Challenges of load balancing algorithms in cloud computing utilizing data mining tools

Anouar Ben Halima, Hafssa Benaboud
Intelligent Processing and Security of Systems, Faculty of Sciences, Mohammed V University in Rabat, Rabat, Morocco

Article Info

Article history:

Received Jun 11, 2024
Revised Dec 10, 2024
Accepted Dec 19, 2024

Keywords:

Cloud computing
Data mining
J48 algorithm
K-means algorithm
Load balancing

ABSTRACT

In the cloud computing environment, load balancing plays an important role in the efficient operation of cloud computing, where a multitude of resources serve diverse workloads and fluctuating demands. In the rapidly evolving cloud computing, efficient resource management, and optimization are critical for maximizing performance, scalability, and cost-effectiveness. Load balancing algorithms aim to distribute workloads across cloud resources to ensure optimal utilization and maintain high availability of services. This paper presents a comparative study of load balancing algorithms in cloud computing using data mining tools. It underscores the complexity of selecting algorithms for effective load balancing in scenarios with diverse criteria, emphasizing its critical importance for future research and practical implementations. The experimental results are presented, evaluating the performance of different load balancing algorithms using data-mining tools. The outcomes highlight the substantial difficulties when building a model with unacceptable errors to cover users' needs while selecting the desired load balancing method.

Corresponding Author:

Anouar Ben Halima
Intelligent Processing and Security of Systems, Faculty of Sciences, Mohammed V University in Rabat
4, Avenue Ibn Battouta, B.P. 1014 RP, Rabat, Morocco
Email: anouar_benhalima@um5.ac.ma

1. INTRODUCTION

The efficiency of cloud computing relies heavily on load balancing, which is a critical operational component that determines various performance metrics such as throughput and response time. Achieving success in cloud computing requires meeting various criteria, driving the need for careful evaluation of many load balancing algorithms and techniques. Among a wide range of algorithms and techniques aimed at solving the complexities of load balancing, satisfactory results remain difficult to obtain, especially in the face of many distinct and evolving criteria. Despite efforts to satisfy specific groups of criteria, finding a comprehensive solution remains a formidable challenge, mainly due to the variability of key criteria across different environments and the complex nature of these criteria.

The most recent literature focuses on citing major techniques in comparative tables and then noting the pros and cons of each algorithm depending on metrics, while other current research classifies load balancing as a taxonomy into hierarchical schemes without utilizing them into insights. Our method refers to a novel technique that utilizes a data mining tool-dependent strategy. The study seeks to address the lacunae within recent literature concerning load balancing algorithms by examining and updating metrics, evaluating prevalent algorithms and techniques, and determining the challenges associated with selecting appropriate load balancing methodologies. A critical focus is on using data mining tools to demonstrate the difficulties inherent in selecting the optimal algorithm that comprehensively satisfies the various criteria essential for effective load balancing. In other words, this study aims to underline the complexity of selecting an appropriate technique with multiple metrics for load balancing using statistical methods and insights. Furthermore, the paper displays the results of developing a model to meet the user's objectives to highlight the core problem of load balancing techniques.

To evaluate load balancing algorithms, the following metrics are useful:
- Throughput: Throughput refers to the total number of tasks successfully completed in a given period. It is obvious that a high throughput is necessary.
- Associated overhead: The total amount of overhead incurred during the load balancing algorithm's execution. For the method to be implemented successfully, it should be minimal overhead.
- Fault tolerant: It is the ability of the algorithm to perform correctly and uniformly even in conditions of failure at any arbitrary node in the system or in the event of a breakdown.
- Migration time: The time duration was taken in migration or transfer of a computational task from one system or environment to another. This time should be minimal to improve the performance of cloud computing.
- Response time: It is the minimum time that a distributed system takes to respond to executing a specific load balancing algorithm.
- Resource utilization: It is the level to which the resources of the cloud are utilized. The most effective load balancing algorithm maximizes the use of available resources.
- Scalability: Scalability determines the ability of the system to accomplish a load balancing algorithm with a limited number of processors or machines.
- Power saving: It represents the mechanism of energy consumption to maintain good quality of service (QoS) of data centers. For example, energy can be conserved by making use of virtual machine (VM) migrations.
- Performance: It represents the effectiveness of the system after performing load balancing. Obviously, if all the above parameters are optimally satisfied, then it highly improves the performance of cloud computing.

This paper is organized as follows. In section 2, authors cite the preceding literature on load balancing algorithms. Section 3 gives the experimental methodology. In section 4, authors give the results including a comprehensive discussion of the outcomes as well as an analysis of the obtained results. In section 5, authors provide limitations and the possibility of extending. Section 6 concludes this paper and gives future research.

2. RELATED WORK

Load balancing algorithms have been the subject of recent studies in the literature. Mishra *et al.* [1] introduces a taxonomy for cloud load balancing algorithms, exploring key performance parameters and their impacts. Performance analysis of heuristic-based algorithms is conducted using the CloudSim simulator, with a detailed presentation of the results. Using the same simulator, Elnagar *et al.* [2] proposes a new algorithm that reduces response time and processing time metrics compared to the common algorithms translation lookaside buffer (TLB), round robin (RR), and approximate maximum load balancing (AMLB). It improves the distribution of tasks between different VMs by reducing the loading gap between the heaviest loaded and the lightest loaded VMs with significant value. Junaid *et al.* [3] propose the data files type formatting (DFTF) load balancing algorithm, integrating a modified cat swarm optimization (CSO) and support vector machines (SVM) classifiers to classify cloud data. Simulation results demonstrate improved performance metrics compared to existing approaches. The review in study [4] aims to critically analyze existing load balancing techniques, discussing parameters like throughput, migration time, and scalability. It highlights the shortcomings of traditional load balancing (LB) algorithms in cloud computing and advocates for integrating fault tolerance (FT) metrics, proposing a novel FT-based LB algorithm to address this need. In the same way, Oyediran *et al.* [5] cite common challenges and benefits of the most common techniques of load balancing. A different approach to studying load balancing in cloud computing involves leveraging software-defined networking (SDN). Yzzogh and Benaboud [6] focuses on recent research highlighting the use of SDN to enhance load balancing in cloud environments. Furthermore, Halima *et al.* [7] gives a comparative study exploring the critical role of predictive load balancing. In the same context, Aron and Abraham [8] discuss the performance of popular load balancing algorithms and techniques. However, while they describe various load balancing schemes with propositions and conditions depending on the cloud environment, they only focus on a few algorithms and overlook some developed ones, such as the honey-bee foraging algorithm [9] and the optimized genetic algorithm [10].

Many algorithms have been implemented, but the results are not efficient because researchers have focused on some metrics while ignoring others. Therefore, all the algorithms, whether mentioned in the cited article or not, treat load balancing as an individual metric. Thus, scientists try to satisfy only one to six metrics at most. Therefore, we should attempt to solve the problem as associated sub-problems of metrics. Unlike the majority of previous studies that mention the load balancing problem, our study uses a novel statistical approach that relies on data mining tools taking into account the metrics of load balancing. In other words, the prior citations were founded on these metrics as comparison research. Conversely, we will address them and take into consideration a new strategy by leveraging these metrics to transform them into relevant information using data mining to determine the complexity for users to select one technique of load balancing.

3. METHOD

Methodologically, this study collects and organizes previous studies into a structured table format before implementing updates due to recent studies. Subsequently, a comparative analysis, employing analytical methodologies and data mining techniques, is undertaken to examine and evaluate these algorithms, thereby demonstrating the challenges inherent in achieving efficient load balancing. Data mining is the process of extracting valuable information and patterns from massive amounts of data. It covers statistical methods as well as collection, extraction, analysis, and statistical techniques. It is also known as the knowledge discovery process, knowledge mining from data, or data/pattern analysis. Data mining is a logical process of locating pertinent information to understand the data. Additionally, the term data mining encompasses many techniques and procedures used to examine and transform data. This paper focuses on two important methods: classification and clustering. To analyze datasets of the criteria cited in Table 1, we introduce some important data mining techniques. These tools help us better understand the problem of the comparative study of the most popular algorithms mentioned in Table 1.

Classification (also known as classification trees or decision trees) is a data mining algorithm that creates a step-by-step guide for determining the output of a new data instance. The tree it creates represents a decision-making process, where each node in the tree represents a spot where a decision must be made based on the input. Moving to the next node depends on the decision, and we continue until we reach a leaf that predicts the output. In our experience, we use J48, which can build a model and create decision trees of data sets based on their attributes. The objective of decision trees is to progressively generalize the decision tree until it reaches a balance between flexibility and accuracy. J48 is an extension of iterative Dichotomizer (ID3) that accounts for missing values, decision tree pruning, continuous attribute value ranges, and derivation of rules.

Clustering is another tool for analyzing data. Given a set of data points, we can use a clustering algorithm to classify each data point into certain groups. K-means is among the most well-known clustering algorithms. That is taught in many introductory data science and machine learning patterns, but the one disadvantage of K-means is that we must choose the value of K before running the algorithm. For our case, K represents the number of data groups created using the criteria keys. Then we vary it to control the number of created groups as needed. In other words, “K” represents the groups of algorithms that have the same criteria attributes. Clustering allows users to make groups of data to determine patterns from the data. Clustering has its advantages when the data set is defined, and a general pattern needs to be determined from the data. One defining benefit of clustering over-classification is that every attribute in the data set is used to analyze the data. A major disadvantage of using clustering is that the user is required to know ahead of time how many groups he wants to create.

To implement these two mentioned tools, we utilize a common tool called WEKA1, which stands for “Waikato environment for knowledge analysis.” WEKA is a collection of machine-learning algorithms for data mining. It includes tools for data preparation, classification, regression, clustering, association rules mining, and visualization. Weka also includes a metric known as squared error, typically used to assess regression models. This metric quantifies the degree to which a regression model accurately fits the data. Lower squared error values indicate superior model performance. Additionally, accurate model evaluation using Weka necessitates the consideration of correctly classified instances.

3.1. Experiments

In this section, a series of experiments have been arranged to investigate the performance characteristics of various algorithms. The algorithms under consideration include: RR, DynamicRR, ShortestJobScheduling, Min-Min, Max-Min, opportunistic load balancing (OLB+), load balancing min-min (LBMM), cost load balancing with virtual machines (CLBVM), predictive adaptive load balancing (PALB), fault-aware min-min load balancing (FAMLB), throttled, HoneyBeeForaging and ActiveClustering. More Algorithms are depicted in Table 1. These algorithms will represent the predicted class of our created model.

Furthermore, the evaluation of these algorithms was conducted based on 9 distinct metrics, namely: performance, throughput, overhead, tolerant, migration time, response time, resource utilization, scalability, and power saving. We ran our experimental study against the 32 algorithms as proof of concept during the training set. These metrics are considered as inputs of our model.

Table 1. Comparative table of the most important algorithms

Algorithm	Performance	Throughput	Overhead	Tolerant	Migration time	Response time	Resource utilization	Scalability	Power saving
RR [11]	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No
DynamicRR [11]	No	Yes	Yes	Yes	Yes	No	Yes	No	No
ShortestJobSheduling [12]	No	No	No	No	No	No	Yes	No	No
Min-Min [13]	Yes	Yes	Yes	No	No	Yes	Yes	No	No
Max-Min [14]	Yes	Yes	Yes	No	No	Yes	Yes	No	No
OLB+LBMM [15]	Yes	No	No	No	No	No	Yes	No	No
CLBVM [16]	Yes	Yes	No	No	No	Yes	Yes	No	No
PALB [17]	No	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
FAMLB [18], [19]	No	Yes	Yes	No	Yes	Yes	Yes	Yes	No
Throttled [20]	Yes	No	No	Yes	Yes	Yes	Yes	Yes	No
HoneyBeeForaging [21]	No	No	No	No	No	No	Yes	No	No
ActiveClustering [22]	No	No	Yes	No	Yes	No	Yes	No	No
BiasedRandomSapmling [23]	Yes	Yes	Yes	No	No	No	No	Yes	No
GeneralizedPriorityAlgo [23]	No	Yes	No	No	Yes	No	Yes	No	No
JoinIdleQueue [24]	Yes	No	Yes	No	No	Yes	No	No	No
GenetecAlgorithm [25]	Yes	No	No	No	No	No	Yes	No	No
AntColony [26]	Yes	No	No	No	Yes	No	Yes	No	No
StochasticHillClimbingTech [27]	Yes	Yes	No	No	No	Yes	Yes	No	No
DecentralizeContentAware [28]	Yes	No	Yes	No	No	Yes	Yes	Yes	No
Server-basedLBForIDServices [29]	Yes	No	No	No	No	Yes	No	No	No
Lock-freeMulti-processing [30]	Yes	Yes	No	No	No	No	No	No	No
Scheduling [31]	No	No	Yes	No	No	No	Yes	No	No
Load balancing virtual storage strategy (LBVS) [32]	Yes	No	No	Yes	No	Yes	No	Yes	No
TaskShedulingbasedOnLB [33]	Yes	No	No	No	No	Yes	Yes	No	No
Ant colony and complex network theory based load balancing (ACCLB) [33]	Yes	No	No	No	No	Yes	Yes	No	No
EventDriven [34]	No	No	No	No	No	No	Yes	Yes	No
CARTON [35]	Yes	No	Yes	No	No	No	Yes	No	Yes
Central load balancer (CAB) [36]	No	No	Yes	No	Yes	No	Yes	No	No
Vector dot [37]	No	No	No	No	No	No	Yes	No	No
Simulated annealing (SA) [38]	No	No	No	No	No	Yes	Yes	No	Yes
Load forecasting and capacity-based (LFCB) [36]	Yes	Yes	No	No	No	No	No	No	No
Global load balancing strategy (GLBS) [39]	Yes	No	No	No	No	Yes	No	No	No

4. RESULTS AND DISCUSSION

4.1. Results

Achieving effective load balancing is a paramount challenge in managing resources within cloud computing environments. Despite implementing various algorithms, the outcomes often fall short of meeting the comprehensive metric outlined in the comparative Table 1. The tabulated data underscores that not all algorithms satisfy all specified metrics. Instances where the table entries are marked with “NO” denote unfavorable outcomes in this metric, while “yes” signifies that the algorithm incorporates this metric. For instance, cost load balancing with dynamic migration (CLBDM) excels in throughput but exhibits shortcomings in terms of speed and complexity. Another illustrative example is LB Min-Max, which meets seven metrics satisfactorily; however, its request time is excessively low, leading to prolonged user wait times for response reception. It is crucial to emphasize that certain metrics are deemed more pivotal for the user’s needs than others in the evaluation process.

Taking this tabulated data, we present the outcomes of our experiments for both classification and clustering. This presentation draws upon the information in Table 1, illustrating the most effective algorithms based on crucial metrics. To elaborate, we take the data from the table and transform it into a standard database format for conducting the experiments. Efforts were directed toward integrating two data mining tools classification and clustering to transform the accumulated data into exploitable information.

The table has thirteen-two rows of data instances and ten metrics as attributes. The initial phase of the experimental process involves delineating the inputs and outputs of the datasets. The data inputs encompass metrics such as scalability, performance, and throughput, while the singular output targeted for prediction pertains to the primary column in the table denoted as "algorithm". This serves as the central quandary that researchers endeavor to address.

4.1.1. Classification

Starting with the classification tool when building a model, the experimental classification shows that twelve data instances (rows) are correctly classified, while twenty are not. The created model has a squared error rate of more than sixteen-two percent. Additionally, the decision tree generated using the J48 algorithm has twelve leaves of algorithms as outcomes, with the "performance" metric attribute represented at the root of the tree, explaining that the flow of decisions must start with the "Performance" metric. Also, the decision tree of the model generates just twelve results that cannot be achieved to cover all the other algorithms mentioned in Table 1. These metrics are represented in gray color in Figure 1, and algorithms are displayed in blue in the same figure. Therefore, the flow of the decision tree has many ambiguities as we can cite; for example, the node mentioned "throughput" led to an "overhead" node even if the decision is "yes" or "no"; thus, they led to the same metric.

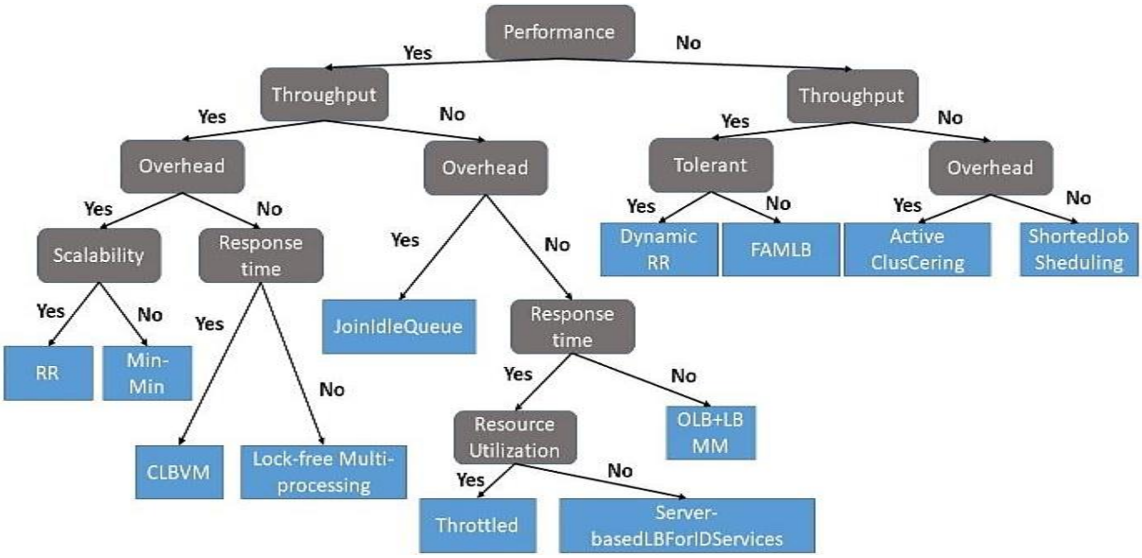

Figure 1. View of the tree using the J48 algorithm method to analyze data instances from the table result

4.1.2. Clustering

For the clustering analysis, we use the K-means algorithm, and thus we vary the value of "K" that remains to several desired algorithms that have the same metric. For instance, with K=5, we obtain a model with a squared error of 42, while for K=10, we obtain ten clustered instances with a squared error equal to 24. As we increase the value of K, the squared error decreases. Finally, for K equal to or greater than 23, the squared error stays zero, which means a clean and accrued model but unlikely with an important value of K that indicates several algorithm groups that we are working to address. Table 2 shows the variation of the squared errors with the different values of K (number of desired algorithms), and the associated graph is shown in Figure 2 represents the variation of squared error during changing K value that controls the instances having the same metrics or in other words, number of algorithms having the same metrics.

Table 2. Variation of squared error depending on K value

Value of K (number of algorithms)	Squared error
2	68
5	42
7	33
10	24
20	5
21	4
32	0

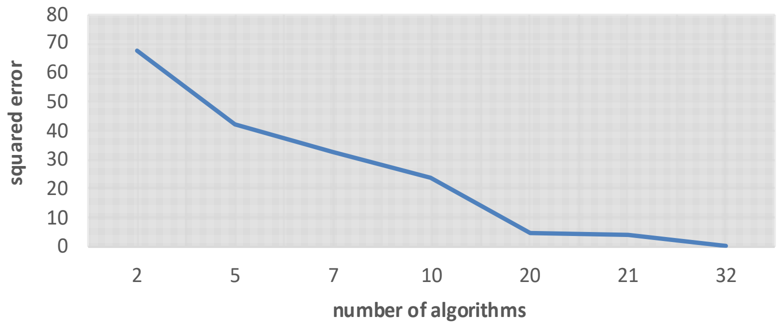

Figure 2. Graph view of the variation of squared errors depending on K (number of algorithms group) value

4.2. Discussion

In general, previous studies have addressed the load balancing issue by discussing the advantages and disadvantages of each technique or by creating a hierarchical taxonomy. Additionally, the prevailing approach in existing studies involves compiling and categorizing algorithms into tables without effectively leveraging them to derive meaningful insights or actionable information. In contrast, our study introduces a new approach to elucidate the complexity of load balancing in choosing the desired technique to achieve users' objectives through data mining tools. To this end, the experimental exploration, encompassing both classification and clustering, revealed pervasive ambiguity in determining the most suitable algorithms to achieve user needs based on various metrics.

4.2.1. Classification

Despite employing multiple metrics for selection, the outcome remained inconclusive, marked by significant errors encountered during the utilization of Weka for experimentation. This suggests that the classification model fails to meet the ambitions of users and researchers to satisfy various metrics due to significant errors during its creation, primarily stemming from inaccurate content. Subsequently, this directly affects users in selecting one or a group of suitable algorithms. Notably, the graphical representation Figure 1 of the decision tree underscores the necessity to initiate the process of selecting metrics predominantly focusing on the "Performance" metric, which might not always align with the comprehensive considerations required for cloud users. Furthermore, the flow of selected metrics presents significant ambiguity in many metrics during the process of selecting the desired load balancing algorithm.

4.2.2. Clustering

Similarly, in the clustering tool, the created clustering model yields poor results, failing to meet the user's objectives. Unfortunately, due to the zeroed value error of the model, the outcomes become illegible despite creating many choices of suitable algorithms. This confusion consistently troubles users when attempting to select the desired algorithms with appropriate metrics. In other words, in the realm of clustering, the model failed to align with our objectives of pinpointing a singular algorithm suitable for clustering. This discrepancy further complicates the selection process.

Moreover, the validation process highlights the inherent challenges in selecting load balancing algorithms, especially when confronted with an expanding array of criteria. This reaffirms the pressing need for a new adaptable model that can seamlessly integrate new updates or additional algorithms, ensuring its continual relevance and usability. This proposed model promises to be a robust framework capable of addressing the complexities inherent in algorithm selection within data mining practices. Therefore, this endeavor unveiled the persistent challenge of selecting an optimal individual or group of algorithms, accentuating the complexities associated with this task. Consequently, a foundational model for data mining was developed, with the potential to evolve and accommodate future updates or the incorporation of new algorithms.

5. LIMITATIONS AND EXTENSION

Our method considers a novel method for load balancing algorithm evaluation by employing data mining tools, which is rare in current literature. We ignore alternative statistical techniques in our experiments and limit ourselves to only two methods: classification and clustering. Similarly, we only provide the most significant load balancing measures rather than all of them because of their complexity and variety. Furthermore, to satisfy the users the parameters and metrics of this model can be developed in the future depending on users of cloud computing.

6. CONCLUSION

Efficient load balancing is essential in cloud computing environments, where diverse workloads and changing demands require optimal resource utilization. Effective resource management and optimization are crucial for enhancing performance, scalability, and cost-effectiveness in this rapidly evolving field. The paper conducted a comparative analysis of load balancing algorithms in cloud computing, employing data mining techniques. It highlighted the challenges in algorithm selection due to varied criteria, emphasizing the need for further research and practical applications. We have shown that achieving complete satisfaction with load balancing algorithms in cloud computing while considering all metrics is difficult. In our future work, we propose a new strategy to achieve well load balancing, by taking advantage of the benefits and drawbacks of such a technique. In addition, we will exploit the created model to help cloud users select suitable metrics.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Anouar Ben Halima	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓			✓
Hafssa Benaboud				✓	✓					✓	✓	✓		

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT *(mandatory)* (10 PT)

Authors state no conflict of interest.

DATA AVAILABILITY

The data that support the findings of this study are available on request from the corresponding author, Anouar Ben Halima. The data, which contain information that could compromise the privacy of research participants, are not publicly available due to certain restrictions.

REFERENCES

[1] S. K. Mishra, B. Sahoo, and P. P. Parida, "Load balancing in cloud computing: a big picture," *Journal of King Saud University - Computer and Information Sciences*, vol. 32, no. 2, pp. 149–158, Feb. 2020, doi: 10.1016/j.jksuci.2018.01.003.

[2] N. G. Elnagar, G. F. Elkabbany, A. A. Al-Awamry, and M. B. Abdelhalim, "Simulation and performance assessment of a modified throttled load balancing algorithm in cloud computing environment," *International Journal of Electrical and Computer Engineering*, vol. 12, no. 2, pp. 2087–2096, Apr. 2022, doi: 10.11591/ijece.v12i2.pp2087-2096.

[3] M. Junaid *et al.*, "Modeling an optimized approach for load balancing in cloud," *IEEE Access*, vol. 8, pp. 173208–173226, 2020, doi: 10.1109/ACCESS.2020.3024113.

[4] M. A. Shahid, N. Islam, M. M. Alam, M. M. Su'ud, and S. Musa, "A comprehensive study of load balancing approaches in the cloud computing environment and a novel fault tolerance approach," *IEEE Access*, vol. 8, pp. 130500–130526, 2020, doi: 10.1109/ACCESS.2020.3009184.

[5] M. O. Oyediran, O. S. Ojo, S. A. Ajagbe, O. Aiyeniko, P. C. Obuzor, and M. O. Adigun, "Comprehensive review of load balancing in cloud computing system," *International Journal of Electrical and Computer Engineering*, vol. 14, no. 3, pp. 3244–3255, Jun. 2024, doi: 10.11591/ijece.v14i3.pp3244-3255.

[6] H. Yzzogh and H. Benaboud, "Using SDN to enhance load balancing in cloud computing: An overview and future directions," in *Proceedings - 6th International Conference on Advanced Communication Technologies and Networking, CommNet 2023*, Dec. 2023, pp. 1–6, doi: 10.1109/CommNet60167.2023.10365294.

[7] A. Ben Halima, H. Yzzogh, and H. Benaboud, "Predictive load balancing in cloud computing: a comparative study," in *ACM International Conference Proceeding Series*, Apr. 2024, pp. 1–6, doi: 10.1145/3659677.3659713.

[8] R. Aron and A. Abraham, "Resource scheduling methods for cloud computing environment: The role of meta-heuristics and artificial intelligence," *Engineering Applications of Artificial Intelligence*, vol. 116, p. 105345, Nov. 2022, doi: 10.1016/j.engappai.2022.105345.

[9] S. Senthilkumar, K. Brindha, P. R. R, and P. Angulakshmi, "Honey-bee foraging algorithm for load balancing in cloud computing optimization," *International Journal of Engineering Science and Computing*, vol. 7, no. 12, pp. 15840–15844, 2017.

[10] F. Nzanywayingoma and Y. Yang, "Analysis of particle swarm optimization and genetic algorithm based on task scheduling in cloud computing environment," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 1, 2017, doi: 10.14569/ijacsa.2017.080104.
[11] N. Pasha, A. Agarwal, and R. Rastogi, "Round robin approach for VM load balancing algorithm in cloud computing environment," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 4, no. 5, pp. 34–39, 2014.
[12] R. K. Mondal, E. Nandi, and D. Sarddar, "Load balancing scheduling with shortest load first," *International Journal of Grid and Distributed Computing*, vol. 8, no. 4, pp. 171–178, Aug. 2015, doi: 10.14257/ijgdc.2015.8.4.17.
[13] T. Kokilavani and D. I. George Amalarethinam, "Load balanced MinMin algorithm for static meta-task scheduling in grid computing," *International Journal of Computer Applications*, vol. 20, no. 2, pp. 42–48, Apr. 2011, doi: 10.5120/2403-3197.
[14] C. Zenon, M. Venkatesh, and A. Shahrzad, "Availability and load balancing in cloud computing," *International Conference on Computer and Software Modeling IPCSIT vol.14 (2011) IACSIT Press, Singapore*, vol. 14, pp. 134–140, 2011, [Online]. Available: https://epress.lib.uts.edu.au/research/handle/10453/19140.
[15] S. C. Wang, K. Q. Yan, W. P. Liao, and S. S. Wang, "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 2010*, Jul. 2010, vol. 1, pp. 108–113, doi: 10.1109/ICCSIT.2010.5563889.
[16] A. Bhadani and S. Chaudhary, "Performance evaluation of web servers using central load balancing policy over virtual machines on cloud," in *COMPUTE 2010 - The 3rd Annual ACM Bangalore Conference*, Jan. 2010, pp. 1–4, doi: 10.1145/1754288.1754304.
[17] G. Michael, K. L. Smith, and S. S. Vrbsky, "Power-aware load balancing for cloud computing," *Lecture Notes in Engineering and Computer Science*, vol. 2193, no. 1, pp. 127–132, 2011.
[18] S. Sethi, "Efficient load balancing in cloud computing using fuzzy logic," *IOSR Journal of Engineering*, vol. 02, no. 07, pp. 65–71, Jul. 2012, doi: 10.9790/3021-02716571.
[19] M. S. Q. Zulkar Nine, M. A. K. Azad, S. Abdullah, and R. M. Rahman, "Fuzzy logic based dynamic load balancing in virtualized data centers," in *IEEE International Conference on Fuzzy Systems*, Jul. 2013, pp. 1–7, doi: 10.1109/FUZZ-IEEE.2013.6622384.
[20] N. Xuan Phi, C. T. Tin, L. N. Ky Thu, and T. C. Hung, "Proposed load balancing algorithm to reduce response time and processing time on cloud computing," *International journal of Computer Networks & Communications*, vol. 10, no. 3, pp. 87–98, May 2018, doi: 10.5121/ijcnc.2018.10307.
[21] M. Randles, D. Lamb, and A. Taleb-Bendiab, "Experiments with honeybee foraging inspired load balancing," in *Proceedings - International Conference on Developments in eSystems Engineering, DeSE 2009*, Dec. 2009, pp. 240–247, doi: 10.1109/DeSE.2009.19.
[22] M. Randles, D. Lamb, and A. Taleb-Bendiab, "A comparative study into distributed load balancing algorithms for cloud computing," in *24th IEEE International Conference on Advanced Information Networking and Applications Workshops, WAINA 2010*, 2010, pp. 551–556, doi: 10.1109/WAINA.2010.85.
[23] D. A. Agarwal and S. Jain, "Efficient optimal algorithm of task scheduling in cloud computing environment," *International Journal of Computer Trends and Technology*, vol. 9, no. 7, pp. 344–349, Mar. 2014, doi: 10.14445/22312803/ijctt-v9p163.
[24] Y. Lu, Q. Xie, G. Kliot, A. Geller, J. R. Larus, and A. Greenberg, "Join-idle-queue: A novel load balancing algorithm for dynamically scalable web services," *Performance Evaluation*, vol. 68, no. 11, pp. 1056–1071, Nov. 2011, doi: 10.1016/j.peva.2011.07.015.
[25] K. Dasgupta, B. Mandal, P. Dutta, J. K. Mandal, and S. Dam, "A genetic algorithm (GA) based load balancing strategy for cloud computing," *Procedia Technology*, vol. 10, pp. 340–347, 2013, doi: 10.1016/j.protcy.2013.12.369.
[26] R. Mishra, "Ant colony optimization: a solution of Load balancing in cloud," *International journal of Web & Semantic Technology*, vol. 3, no. 2, pp. 33–50, Apr. 2012, doi: 10.5121/ijwest.2012.3203.
[27] B. Mondal, K. Dasgupta, and P. Dutta, "Load balancing in cloud computing using stochastic hill climbing: a soft computing approach," *Procedia Technology*, vol. 4, pp. 783–789, 2012, doi: 10.1016/j.protcy.2012.05.128.
[28] H. Mehta, P. Kanungo, and M. Chandwani, "Decentralized content aware load balancing algorithm for distributed computing environments," in *International Conference and Workshop on Emerging Trends in Technology 2011, ICWET 2011 - Conference Proceedings*, 2011, pp. 370–375, doi: 10.1145/1980022.1980102.
[29] A. M. Nakai, E. Madeira, and L. E. Buzato, "Load balancing for internet distributed services using limited redirection rates," in *Proceedings - 2011 Latin-American Symposium on Dependable Computing, LADC 2011*, Apr. 2011, pp. 156–165, doi: 10.1109/LADC.2011.25.
[30] X. Liu, L. Pan, C. J. Wang, and J. Y. Xie, "A lock-free solution for load balancing in multi-core environment," in *2011 3rd International Workshop on Intelligent Systems and Applications, ISA 2011 - Proceedings*, May 2011, pp. 1–4, doi: 10.1109/ISA.2011.5873313.
[31] J. Hu, J. Gu, G. Sun, and T. Zhao, "A scheduling strategy on load balancing of virtual machine resources in cloud computing environment," in *Proceedings - 3rd International Symposium on Parallel Architectures, Algorithms and Programming, PAAP 2010*, Dec. 2010, pp. 89–96, doi: 10.1109/PAAP.2010.65.
[32] H. Liu, S. Liu, X. Meng, C. Yang, and Y. Zhang, "LBVS: a load balancing strategy for virtual storage," in *Proceedings - 2010 International Conference on Service Science, ICSS 2010*, 2010, pp. 257–262, doi: 10.1109/ICSS.2010.27.
[33] Y. Fang, F. Wang, and J. Ge, "A task scheduling algorithm based on load balancing in cloud computing," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6318 LNCS, no. M4D, Springer Berlin Heidelberg, 2010, pp. 271–277.
[34] Z. Zhang and X. Zhang, "A load balancing mechanism based on ant colony and complex network theory in open cloud computing federation," in *ICIMA 2010 - 2010 2nd International Conference on Industrial Mechatronics and Automation*, May 2010, vol. 2, pp. 240–243, doi: 10.1109/ICINDMA.2010.5538385.
[35] V. Nae, R. Prodan, and T. Fahringer, "Cost-efficient hosting and load balancing of massively multiplayer online games," in *Proceedings - IEEE/ACM International Workshop on Grid Computing*, Oct. 2010, pp. 9–16, doi: 10.1109/GRID.2010.5697956.
[36] N. J. Kansal and I. Chana, "Cloud load balancing techniques: a step towards green computing," *International Journal of Computer Science Issues*, vol. 9, no. 1, pp. 238–246, 2012.
[37] S. Sidhu and M. Mittal, "A new era to balance the load on cloud using vector dot load balancing method," *International Journal of Technology and Computing*, vol. 2, no. 4, pp. 413–418, 2016.
[38] S. Sreenivasamurthy and K. Obraczka, "Clustering for load balancing and energy efficiency in IoT applications," in *Proceedings - 26th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS 2018*, Sep. 2018, pp. 319–332, doi: 10.1109/MASCOTS.2018.00038.
[39] V. K. Reddy, K. Deva Surya, M. Sai Praveen, B. Lokesh, A. Vishal, and K. Akhil, "Performance analysis of load balancing algorithms in cloud computing environment," *Indian Journal of Science and Technology*, vol. 9, no. 18, May 2016, doi: 10.17485/ijst/2016/v9i18/90697.

BIOGRAPHIES OF AUTHORS

Anouar Ben Halima received a bachelor's degree in mathematics and computer science from Abdelmalek Essaadi University of Tetouan in 2013. He received a master's degree in computer science engineering from the University of Abdelmalek Essaadi of Tetouan in 2016. Currently, he is a Ph.D. student at Mohamed V University in Rabat, Morocco. His research interests include cloud computing technologies, machine learning, and algorithm optimization. He can be contacted at email: anouar_benhalima@um5.ac.ma

Hafssa Benaboud received her Ph.D. degree in computer sciences from Burgundy University Dijon-France in 2004. In 2005, she joined as an assistant professor at Applied Sciences National School (ENSA) of Tangier, Morocco, and has been working as a full professor since 2011 in the Department of Computer Sciences at Mohammed V University in Rabat, Morocco. She has authored more than 30 articles published in international journals and international conference proceedings. Her research interests include network protocols, network security, internet of things, traffic analysis and quality of services. She can be contacted at email: hafssa.benaboud@fsr.um5.ac.ma.