Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Edge Load Balancing by Artificial Immune and LSTM Model for IOT Device Scheduling

PhD Scholar, AtulGarg
Department of CSE, RNTU
Dr. PritajYadav
Associate Professor Department of CSE, RNTU

Abstract—IOT device based network communication improves monitoring, reporting, business, lifestyle, etc. This directly increase load on servers, cloud hence a mediator was involved as edge computing. But large number of devices need job completion hence Edge load balancing models were required. Many of existing models provides static and dynamic solutions but none of them learn from previous job sequence. So this paper proposed a model that finds the job sequence for the IOT devices by use of modified artificial immune bio-inspired algorithm. As bio-inspired not need any prior information or training for finding the solution so this fit for the dynamic load balancing. Further paper has utilized job sequence and edge features to train LSTM model for getting initial job sequences of artificial immune algorithm. Experiment was done on different network condition by varying number of edge and IOT devices. It was found that proposed Dynamic Learning Model of Edge Load Balancing (**DLM-ELB)** model has reduces the makespan time of job by 1.403% and improve edge utilization by 2.22% as compared to comparing models.

Index Terms— Load Balancing, Edge Node Computing, Cloud Computing, Genetic Algorithm.

I. INTRODUCTION

A growing number of applications are now being routinely installed on mobile devices, making these devices the primary tools for multimedia consumption, creation, computing, and human-computer interaction. With the rise of mobile cloud computing, there has been a significant increase in the expectations for optimal and reliable services and support for mobile users. The extensive array of cloud resources and services has facilitated the development of numerous innovative applications tailored for smart environments [1]. However, despite the advancements, the current state-of-the-art in mobile cloud computing presents challenges, particularly for communication-intensive applications that demand low latency. This issue becomes even more critical in the context of smart cities or within the Internet of Things (IoT) ecosystem, where rapid and efficient data processing is essential [2]. The traditional cloud computing model struggles to meet the requirements for low latency, location awareness, and mobility support effectively.

The limited size, battery capacity, energy consumption, and inherent latency of mobile devices pose significant challenges when it comes to handling computationally intensive tasks. To mitigate these issues, the concept of mobile edge computing (MEC) has emerged. MEC, also known as multi-access edge computing, is a cutting-edge cloud computing paradigm that brings cloud resources closer to the network's edge, rather than depending on distant central cloud data centers [3]. This approach is designed to minimize latency and improve the efficiency of the overall network architecture.

Edge computing shifts cloud computing applications, data, and services from centralized nodes to the network's periphery, positioning itself between end-user devices and traditional cloud data centers [4]. This strategic placement enables edge computing to handle low-latency and real-time tasks with greater efficiency. By moving the cloud closer to the end-user, edge computing significantly reduces latency [5]. However, despite these advantages, inefficient task allocation can result in an uneven load distribution across nodes. Additionally, the diversity and heterogeneity of edge computing nodes present challenges for directly applying traditional load balancing algorithms to edge environments. As a result, load balancing in edge computing has become a critical focus of academic research.

Load balancing strategies in edge computing are generally classified into two categories: static and dynamic. Static load balancing algorithms distribute the load without considering the previous state of a node, making them suitable in scenarios where load variation across nodes is minimal [6]. However, this makes them unsuitable for the dynamic nature of edge environments. On the other hand, dynamic load balancing strategies, such as task allocation based on intermediary nodes, consider the previous state of nodes when distributing the load. This dynamic approach is more appropriate for edge computing [7], as it can adapt to the varying conditions and demands of the network, ensuring a more balanced and efficient distribution of tasks.

Problem Identify This paper identifiessome set of existing issues of edge load balancing models/algorithms and resolve then by the proposed model.Few of existing approach uses sequential methods to allot edge resource. This should be dynamic.It was found that scheduling should be adaptive if number of edges get increase or number of IOT devices get increases.

Objective This model overcome a job load balancing model for edge computing environment. To develop model that provides low-latency services for terminals. To schedule task dynamically with any number of task or number of edges present in the network.

Rest of paper was structure in four more section where summary of various research work was detailed in second section. Proposed model of edge load balancing was detailed in the third section. Experimental results and comparisons of proposed model was detailed in fourth section of paper. Finally paper was concluded with future scope of work.

II. Related Work

A. Mahapatra et al. [8] explore the challenge of task migration in Fog-Cloud systems, with a particular focus on achieving load balancing to reduce latency, energy consumption, and service time while optimizing resource utilization in latency-sensitive applications. Their research introduces a Fuzzy logic-based algorithm that selects the most suitable layers for offloading tasks by taking into account resource heterogeneity and various system parameters, such as network bandwidth, task size, resource usage, and latency sensitivity. They also propose a Binary Linear-Weight JAYA (BLWJAYA) task scheduling algorithm, which is designed to efficiently allocate incoming IoT requests to computation-rich Fog nodes or virtual machines.

In their work, ZhenhuaDuan et al. [9] present a three-layer mobile, envisioned as a cloudlet architecture. The proposed model stands out for its efficiency in service lookup, scalability, and system stability. A key feature of this model is its load balancing mechanism, which ensures that loads are evenly distributed as the number of MEC servers and query loads increase, thereby maintaining optimal system performance.

Hoseiny et al. [10] propose a task scheduling technique that aims to minimize total computation time and energy consumption in a heterogeneous fog-cloud computing environment. This approach considers specific task characteristics to determine the most appropriate computing environment for each task. However, the method does not address interdependencies between tasks within an application, and the use of metaheuristic algorithms requires manual configuration of scheduling rules, limiting its flexibility in adapting to dynamic computing environments.

Ali et al. [11] introduced a technique based on the Non-dominated Sorting Genetic Algorithm II (NSGA2) to optimize task scheduling within heterogeneous fog cloud computing environments. This method seeks to minimize overall computation time and system costs by dynamically assigning appropriate resources to predefined tasks, framing the task scheduling issue as an optimization problem. However, the reliance on metaheuristic algorithms means the approach necessitates some prior knowledge of the tasks being submitted, limiting its effectiveness in dynamic and complex scenarios where task information is unpredictable.

Ajay Jangra et al. in [12] developed a load balancing framework aimed at resource scheduling in cloud-based healthcare systems. This framework incorporates reinforcement learning algorithms, including Genetic Algorithms (GA), State-Action-Reward-State-Action (SARSA), and Q-learning, to predict optimal solutions for load management. By utilizing these advanced learning techniques, the framework strives to enhance resource scheduling efficiency and maintain balanced task distribution in healthcare environments.

Baburao et al. [13] introduced an model that uses PSO to address load balancing challenges in fog-cloud computing environments. EDRAM aims to decrease task waiting times, reduce latency, and optimize network bandwidth usage while enhancing the Quality of Experience for end-users. This method achieves effective resource allocation by efficiently managing RAM, specifically by clearing out inactive and unused services, thereby maximizing the use of available resources.

In [14], a novel IoT-Edge scheduling strategy grounded in game theory is proposed to address resource allocation and task scheduling challenges within IoT environments. This approach considers two distinct scenarios. In the first scenario, an algorithm is introduced that allows IoT and edge nodes to evaluate each other based on metrics such as delay and resource consumption, facilitating more informed decision-making. The second scenario involves both centralized and distributed scheduling techniques, utilizing the concept of matching to enhance resource allocation efficiency. Furthermore, a model was presented, focusing on achieving stable and efficient resource allocation by considering the preferences of the participating nodes.

III. Proposed Methodology

In this section of paper proposed Dynamic Learning Model of Edge Load Balancing (**DLM-ELB)**was detailed. Each edge node is initialize in the network, with resources. This initialize to allocation by proposed model is shown in fig. 1. Each block in fig 1 was detailed in the section, while notations were summarized in table 1. Dynamic IOT job sequence was manage by the Artificial Immune Genetic Algorithm. This work learn job sequences and available resources to generate initial population as well. For leaning random forest was used.

Table 1 Notation used in Edge Load balancing model DLM-ELB.

Notation	Meaning
E	Edge
IJ	IOT Job

Rm	Edge Resource Memory
Rp	Edge Resource Processor
Rb	Edge Resource Bandwidth
JSP	Job Sequence Population
JS	Job Sequence
A	Affinity of JS
Ne	Number of edges
Nj	Number of Jobs
MST	MakeSpan Time

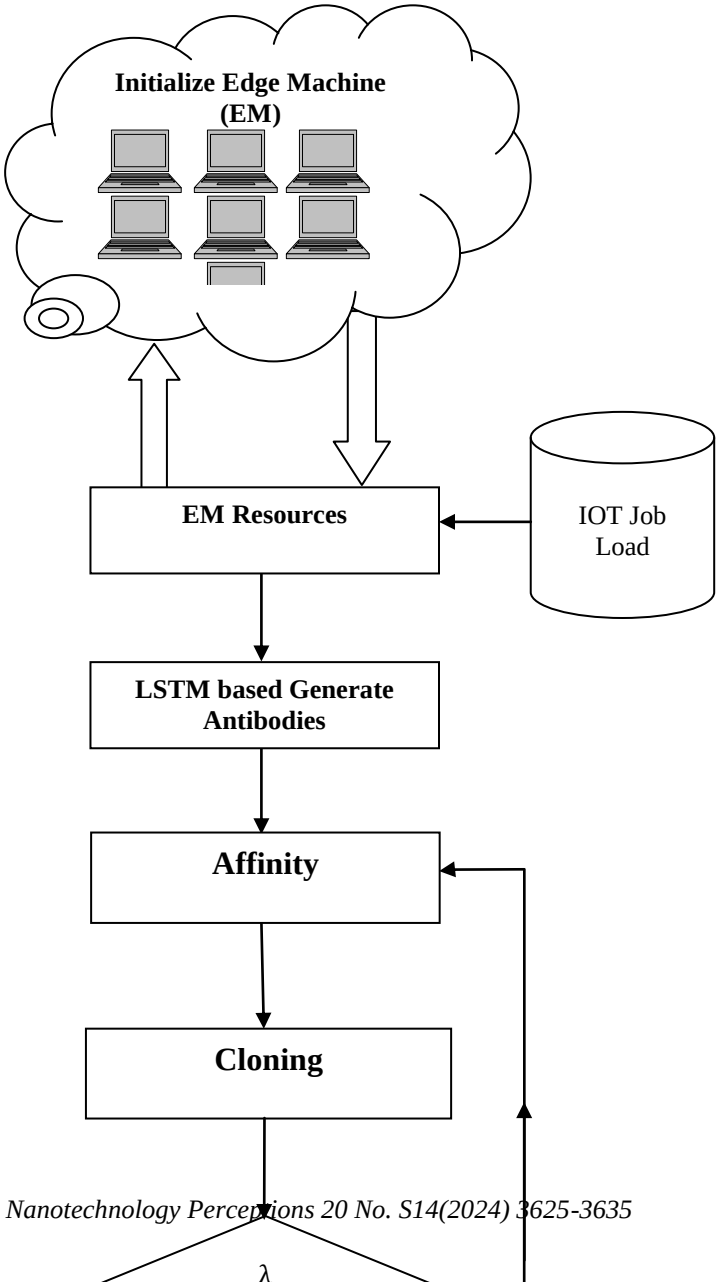

Fig.1Proposed Edge load balancing model.

Edge: Each edge device in a network have its own set of resources Rm Edge Resource Memory, Rp Edge Resource Processor, Rb Edge Resource Bandwidth. So this paper consider that network has ne number of edges to handle Nj number of IOT jobs. Finding the IOT job sequence that will reduces the makespan time is done by the model.

LSTM Learning: Various job requirements were generate for the IOT devices and sequence for the edge was generate for the same. LSTM model get train for generating the IOT device job sequence generation. Trained model was used for the population generation of genetic algorithm. Many of genetic approach uses Gaussian function for population generation but this model uses the previous sequence learning for finding the job pattern. LSTM training vector consist of IOT resources information like processor, memory, bandwidth, etc. apart from this average resource available to allot by edges. These two different type of feature train the model.

Generate Antibodies IOT device job sequence obtained from the LSTM model in order to get variety of sequence job features were shuffled.

$$JSP \leftarrow LSTM\_Antibody(IJ, E)$$

Affinity Affinityis fitness value of probable antibodies. The makespan of the sequence/antibody is term as affinity. JS having minimum makspan is consider as best antibody in the current iteration. Hence affinity A of the JS is estimate by:

Input: JSP, IJ
Output: A // A is Affinity that is MakeSpan of JS
Loop 1:a // a: number of antibodies in JSP
    JS←JSP[a]
    MST←[]
    Loop 1:Nj
        Loop 1:Ne
            MST←Execute_Job(IJ(1,Nj), JS(1,Ne))
        EndLoop
    EndLoop
    A[a]←Sum(MSP)
EndLoop

Cloning is performed based on the affinity value of each antibody within the population, leading to the identification of the optimal solution, JSb. Once the best antibody, JSb, is determined, certain features within the associated set are randomly altered. By change in edge sequence of non best solution as per best solution cloning of the model is done.

$$JSP \leftarrow Cloning(JS_b, JSP)$$

Update Population Affinity and cloning operation is done for n time and after each iteration cloned new JS fitness is estimate by algorithm 1. If new JS is better than its parent then new JS replace population JS otherwise parent is continue. After fix number of t iteration best affinity JS is consider as final solution of the model.

Proposed Algorithm
Input: IJ, E
Output: JS_b
$[R_mR_pR_b]$←Get_Edge_Resources(E)
JSP←LSTM_Antibody(IJ, E)
    Loop 1:t
        A←Affinity(JSP, IJ, $R_mR_pR_b$)
        JS_b←Min(A)
        JSP←Cloning(JS_b, JSP)
        A←Affinity(JSP, IJ, $R_mR_pR_b$)
        JSP←Update_Population(A,JSP)
    EndLoop
A←Affinity(JSP, IJ, $R_mR_pR_b$)
JS_b←Min(A)

Above algorithm reflects the steps of proposed Dynamic Learning Model of Edge Load Balancing (DLM-ELB). It was found that once LSTM model get trained then chance of getting a good job sequence for edge load balancing is very high.

IV. Evaluation Parameters

Experiment work was done on different environmental conditions, where number of edges, and IOT job sequence get modified. Implementation of model was done on MATLAB software and machine having a configuration of processor I312th generation with 8GB RAM. Comparison of Dynamic Learning Model of Edge Load Balancing (DLM-ELB) model was done by existingPreference Based Stable Matching model (PBSM) proposed in [14].

Results

Table 2 Makespan based comparison of edge load balancing models.

Edges	IOT Jobs	DLM-ELB	PBSM
10	20	141.371	142.9045
10	40	155.1015	156.604
10	60	154.159	157.591
10	80	168.2679	170.5323
10	100	156.0484	157.14
20	100	155.0197	157.54
30	100	154.7986	158.808
40	100	154.6159	157.6491
50	100	155.8931	156.3586

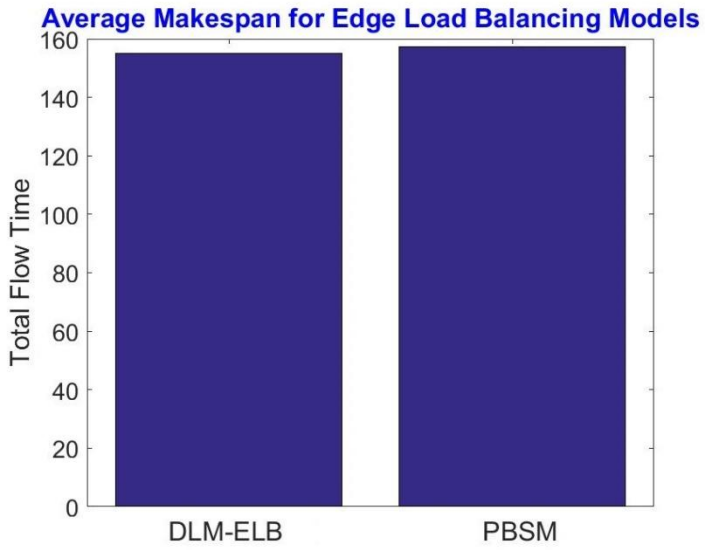

Fig. 2 Average makespan for comparing models.

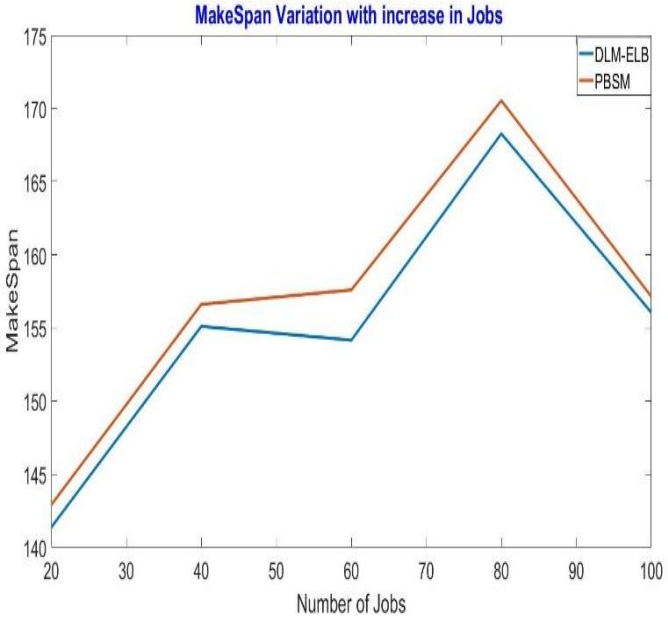

Fig. 3 Makespan variation as per number of IOT jobs.

Table 2 shows IOT device jobs makespan time for the edge load balancing algorithm. It was found that by increase in number of IOT devices in network makespan time of both models get increase. Fig. 2 shows that DLM-ELB model has reduces the makespan time of the model by 1.403% as compared to PBSM. Fig. 3 shows that use of artificial immune algorithm for finding the edge job sequence in dynamic environment works well.

Table 3Total Flow Time based comparison of edge load balancing models.

Edges	IOT Jobs	DLM-ELB	PBSM
10	20	2827.4	2858.1
10	40	6204.1	6264.2
10	60	9095.4	9455.5
10	80	13461	13643
10	100	15605	15713.5

20	100	15502	15754
30	100	15480	15881
40	100	15462	15765
50	100	15589	15636

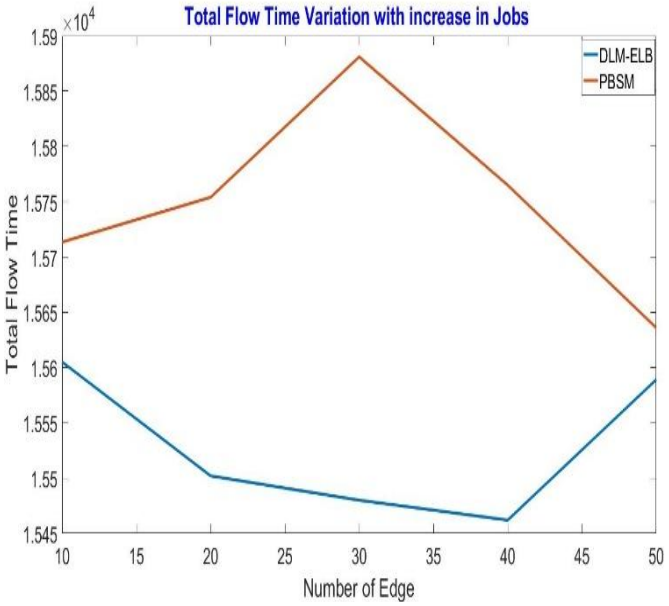

Fig. 4 Total flow time value variation as per number of edges.

Table 3 shows flow time of the edge load balancing comparing models. It was found that with increase in number of edges in the network flow time of the models almost same but with increase in number of IOT jobs total flow time increases. Fig. 4 shows that use of LSTM model for the learning of job sequence has reduces the work flow time by 1.57% as compared to existing PBSM.

Table 4Edge Utilization based comparison of edge load balancing models.

Edges	IOT Jobs	DLM-ELB	PBSM
10	20	74.5	69.5
10	40	87.25	84.75
10	60	93.0508	89.833
10	80	93.625	92.375
10	100	95.9	95
20	100	89.95	89.9
30	100	85.9667	85
40	100	81	80
50	100	73.98	72

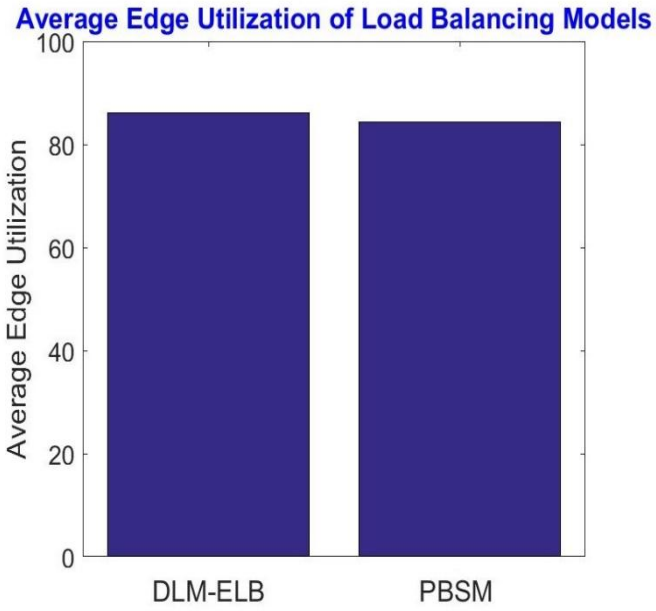

Fig. 5 Average edge utilization of comparing models.

Each edge has different set of resources in the network, hence load balancing model should utilize it for max to improve the network performance. Table 4 and fig. 5 shows that edge utilization was enhanced by the DLM-ELB model as compared to existing PBSM by 2.22%. Use of edge resource feature for the training and testing of LSTM has improved the job sequence generation.

Table 5 Execution time based comparison of edge load balancing models.

Edges	IOT Jobs	DLM-ELB	PBSM
10	20	0.0837	0.0945
10	40	0.0728	0.1088
10	60	0.0892	0.0874
10	80	0.0539	0.0776
10	100	0.06142	0.0643
20	100	0.05481	0.0839
30	100	0.0938	0.0664
40	100	0.0673	0.0755
50	100	0.063	0.0928

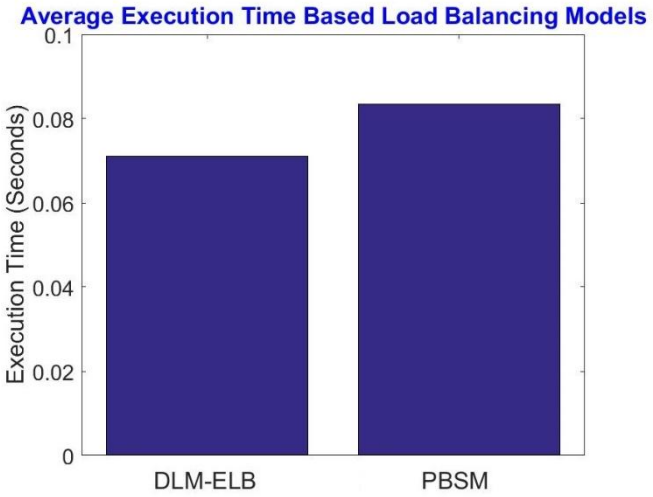

Fig. 6 Average execution time of comparing models.

Table 5 shows IOT device jobs sequence generation execution time for the edge load balancing algorithms. Fig. 6 shows that DLM-ELB model has reduces the makespan time of the model by 14.81% as compared to PBSM. This shows that use of artificial immune algorithm for finding the edge job sequence in dynamic environment works well.

V. Conclusion

This paper proposes a novel model that determines the optimal job sequence for IoT devices using a modified artificial immune bio-inspired algorithm. Bio-inspired algorithms are particularly suitable for dynamic load balancing because they do not require any prior information or training to find solutions. This characteristic makes them ideal for handling the ever-changing conditions typical of IoT environments. In addition to utilizing the bio-inspired algorithm, the paper incorporates job sequence and edge features to train a Long Short-Term Memory (LSTM) model. This LSTM model helps generate initial job sequences for the artificial immune algorithm. The research involved conducting experiments under various network conditions by varying the number of edge devices and IoT devices. The results demonstrated that the proposed Dynamic Learning Model of Edge Load Balancing (DLM-ELB) effectively reduces the total flow time of jobs by 1.57% and improves edge utilization by 2.22% compared to other existing models. These improvements highlight the efficiency and effectiveness of the DLM-ELB model in optimizing job sequences and load balancing in IoT environments.

References

1. PhruksaphanratBPanjavongroj S(2023)A new logarithmic fuzzy full consistency method for prioritizing critical success factors of technology start-ups in ThailandApplied Soft Computing10.1016/j.asoc.2023.110691146:COnline publication date: 1-Oct-2023

2. Chen, W.; Zhu, Y.; Liu, J.; Chen, Y. Enhancing Mobile Edge Computing with Efficient Load Balancing Using Load Estimation in Ultra-Dense Network. Sensors 2021, 21, 3135.

3. Mukherjee, A., De, D., Ghosh, S.K., Buyya, R. (2021). Introduction to Mobile Edge Computing. In: Mukherjee, A., De, D., Ghosh, S.K., Buyya, R. (eds) Mobile Edge Computing. Springer, Cham.

4. S. Singh, in 2017 International Conference on Big Data, IoT and Data Science (BID). Optimize cloud computations using edge computing, (2017), pp. 49–53.

5. X. Xu, Y. Xue, L. Qi, Y. Yuan, X. Zhang, T. Umer, S. Wan, An edge computing-enabled computation offloading method with privacy preservation for internet of connected vehicles. Future Gener. Comput. Syst.96:, 89–100 (2019).

6. T. Zhu, T. Shi, J. Li, Z. Cai, X. Zhou, Task scheduling in deadline-aware mobile edge computing systems. IEEE Internet Things J., 1–1 (2018).

7. L. Yu, L. Chen, Z. Cai, H. Shen, Y. Liang, Y. Pan, Stochastic load balancing for virtual resource management in datacenters. IEEE Trans. Cloud Comput.PP(99), 1–1 (2016).

8. A.Mahapatra, S. K. Majhi, K. Mishra, R. Pradhan, D. C. Rao and S. K. Panda, "An Energy-Aware Task Offloading and Load Balancing for Latency-Sensitive IoT Applications in the Fog-Cloud Continuum," in IEEE Access, vol. 12, pp. 14334-14349, 2024.

9. ZhenhuaDuan, Cong Tian, Nan Zhang, Mengchu Zhou, Bin Yu, Xiaobing Wang, JiangenGuo, Ying Wu. "A novel load balancing scheme for mobile edge computing", Journal of Systems and Software, Volume 186, 2022.

10. F. Hoseiny, S. Azizi, M. Shojafar, F. Ahmadiazar, R. Tafazolli, PGA: A Priority-aware Genetic Algorithm for Task Scheduling in Heterogeneous Fog-Cloud Computing, in: Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2021, pp. 1–6.

11. Ali I.M., Sallam K.M., Moustafa N., Chakraborty R., Ryan M., Choo K.-K.R. An automated task scheduling model using non-dominated sorting genetic algorithm II for fog-cloud systems IEEE Trans. Cloud Comput., 10 (4) (2022), pp. 2294-2308.

12. Ajay Jangra, NeerajMangla. "An efficient load balancing framework for deploying resource schedulingin cloud based communication in healthcare", Measurement: Sensors, Volume 25, 2023.

13. D. Baburao, T. Pavankumar, C.S.R. Prabhu Load balancing in the fog nodes using particle swarm optimization-based enhanced dynamic resource allocation method Appl. Nanosci. (2021), pp. 1-10.

14. A. Bandyopadhyay et al., "EdgeMatch: A Smart Approach for Scheduling IoT-Edge Tasks With Multiple Criteria Using Game Theory," in IEEE Access, vol. 12, pp.