Article

JOURNAL OF UNIVERSITY OF BABYLON

For Pure and Applied Sciences (JUBPAS)

Vol.32; No.3.| 2024

Minimizing Costs in Fog Nodes Through Effective Resource and Task Scheduling

Zaid Ibrahim Rasool[1]

[1]Computer Techniques Engineering Department, Al- Mustaqbal University, 51001, Hilla, Iraq

.

***Corresponding author email**: zaid.ibrahim@uomus.edu.iq

Accepted: 20/9/2024 **Published:** 30/9/2024

ABSTRACT

Background:

This system comprises the fog control node, access node, and Internet of Things (IoT) application. This study aims to reduce scheduling-related cloud costs. A user scheduling method and an economical task are carried out for this reason. The algorithm for job scheduling is constructed to determine the best strategy to handle several jobs in a fog access node. Reallocation technology ultimately reduces delays in both time and service. For analytical objectives, extensive simulations were run and performance metrics were compared with those of other currently in use methods. It was discovered that the suggested technique lowers task latency and offers extreme cost-effectiveness by improving the concurrent capability in the fog node, users, and task scheduling with improved performance statistics that made it possible.

Materials and Methods:

The research paper explores fog computing, a three-layered architecture consisting of IoT devices, fog computing, and cloud computing. The fog computing layer sits between cloud networks and Internet of Things (IoT) devices, providing fast access to these devices and handling computation locally at the network edge.

Results:

The research paper focuses on reducing cloud costs in scheduling. Although the proposed system showed many similarities in execution times compared to the diverse early completion-time method, it consistently lags behind the heterogeneous method by 5 to 10%.

Conclusion:

This research aims to reduce the cloud scheduling costs for Internet of Things (IoT) devices. A user-based scheduling algorithm and economical task allocation are implemented. A constraint-based and user-defined strategy is recommended.

Keywords: Task scheduling, cloud, fog, IoT, and execution time .

مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل للعلوم الصرفة والتطبيقية

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

INTRODUCTION

Because cloud computing provides services via the internet, it lessens the need of users to plan, provides an adaptable computing model, and fulfills business objectives, its emergence has grown significantly and is becoming more and more alluring for both industry and academia [1]–[3]. Many studies indicate that the Internet of Things (IoT) industry would expand quickly due to recent technology breakthroughs like blockchain, and (IoT) and cyber-physical systems (CPS) utilized in an industrial setting[29]. According to estimates, there will be 64 billion IoT devices in use by 2025 [4]. Through the Internet of Things, internet access is made available to everything that a person may need for their typical daily routine, including laptops and mobile phones[3,5]. The need for cloud computing is growing as a result. A computer approach known as "cloud computing" allows users to conveniently and instantly access information or data from a collection of configurable computing resources via the internet [4,6]. Another subset of Internet computing is cloud computing, where all data is kept online or on the cloud. Through the network layer and the internet, the user or client application can access these data. Many devices such as smartphones and tablets benefit financially and in terms of time by having scalability and high-performance capabilities because they increase storage of data capacity and facilitate server-side problem-solving [7]. The Internet of Things gadget application growth results in an increase in service requests and responses, which are inefficient in terms of time, money, and resources when it comes to cloud computing. A problem in internet transmission was ensuring consistent and dependable service. However, the use of a router, gateway, and workstation allowed it to grow[8]. By placing an intermediary between the two, the problem between Internet of Things devices and the Internet of Things has been resolved. The fog server functions as a cache or proxy when a user application requests data since it stores the commonly accessed data locally. With this approach, updated and sophisticated problems will be sent to the cloud for processing, while often-encountered and basic Issues are manageable on the fog node. They reduce the time of requests for services from the handheld device to the internet and manage the massive application successfully, as demonstrated in Figure 1. This makes it very evident that fog nodes are just cloud proxies and are not meant to take the purpose of cloud services and storage. [9, 10].

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

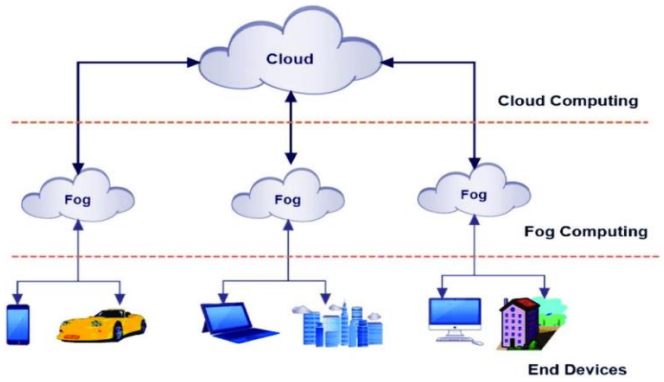

Figure 1. Fog computing model [11]

Numerous dependent or dependent modules that operate in parallel or sequential order to accomplish the goal are components of large-scale applications [12,13].

The heterogeneous distributed system has a similar issue. The temporal complexity of problem-solving increases in scenarios where numerous systems operate as modules for a single main application and take advantage of distributed computing resources [14]. Many research investigations have demonstrated that scheduling algorithms can minimize time; however, this system needs a lot of resources, which is not a problem for a distributed system. However, while assigning tasks in the computing paradigm, it is important to consider that Cloud services are expensive, especially cloud storage [15]. As it has been seen in Figure 2, in fog computing, task scheduling aims to cut down on time by making the most use of cloud resources. This study examines a distributed computing platform that combines cloud computing and fog for workflow-oriented applications. The cloud node is extended by fog nodes installed at the gateway and router. The reallocation approach reduces task latency by optimizing the fog node's concurrent activity.

The structure of this document is as follows. In Section 2, the problem statement is explained. The suggested algorithm and the system design are shown in Section 3. The analysis's findings and the experimental setting are shown in Section 4. The findings and discussion are presented in Section 5, and the conclusion is given in Section 6.

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

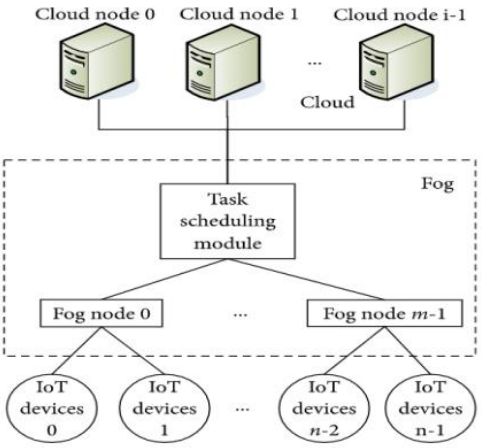

Figure 2. Task scheduling model in fog computing [16]

ASSESSMENT OF LITERATURE

A heterogeneous dispersed system functions as distinct modules for a single primary application that is concurrently running in multiple places [14,15]. Numerous studies have demonstrated that scheduling algorithms can minimize time; however, this system requires a high resource allocation, which is not an issue in a distributed system [17]. Depending on the heterogonous system, Topcuoglu devised an algorithm with heterogonous finishing early in time. Task prioritization and processor selection are its two stages [18]. An approach for the estimated earliest finish time was proposed by Barbosa. In order to maintain the optimal cost table, this algorithm counts the number of processor and tasks [19]. Additionally, Wang suggested a method for the workflow module's time-efficient heterogonous system. This optimization technique is insertion-based. These methods were only made pan-effective rather than cost-effective when they were examined for heterogonous systems [14]. Panda suggested a profit-based job scheduling method that cuts down on both time and cloud service usage. It does not, however, take cloud resource costs into account [13]. Den utilizes hybrid, private, and public clouds with task scheduling algorithms. Each task's priority is set by the deadline restrictions that are specified for each application. This algorithm's limitation was its specific service orientation for both private and public clouds. Every assignment was given a sub-deadline, which is a novel concept. It gets moved from the public to the private cloud based on the deadline. Nevertheless, the trade-off between cost and makespan is not addressed in this study [20]. The network itself and its edges, such as routers and gateways, have been the focus of numerous recent research projects [21]. Bonomi researched the use of fog nodes and their difficulties in cloud integration. The integration of fog computing and IoT was also covered in this study [22]. Alsaffar put up a suggested architecture for allocating

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

resources and assigning IoT services. utilizing the virtual machine's capacity, size, time, and service request characteristics [23]. Souza investigates the issue of service quality when cloud computing is combined with other cloud computing platforms [24]. Simplifying the fog computing task's scheduling problem was the focus of several studies. In his research, Zeng put up a design for fog computing's resource and time management [25]. An alternative approach put forth by Nan uses an adaptive algorithm to make judgments in the fog computing three-tier architecture, and it operates on average time [26]. Nevertheless, fog-based research did not consider the workflow model applications. The application of process models is taken into consideration with fog cloud integration in the suggested study[27,28]. To maximize the compromise between makespan & resource cost, a heuristic technique is recommended taking into account the workflow architecture & deadline constraints [29].

Problem Analysis

Fog nodes usually process information more slowly than cloud services because fog nodes lack the RAM, storage, and computational power typical of the cloud. On the other hand, because cloud resources are expensive, transferring an assignment to the cloud node produces in a greater resource cost. As a result, an unmanaged job assignment method degrades fog-cloud infrastructure performance.

Analysis of task scheduling model

An extensive application using a process architecture is demonstrated Figure 3. To reduce reliance on a single module, a task of this type is broken out into multiple modules rather than just one. Nevertheless, in a dependent job, parallel programming could be impacted by user-defined inputs or the dependence of one module on the result of another.

To keep the application's temporal complexity manageable, many processors are tasked with handling it. This results in an increase in resource costs, which is reasonable for some application infrastructures. However, because cloud computing involves expensive resources, to reduce the cost of cloud utilization, a fixed processor count is used.

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

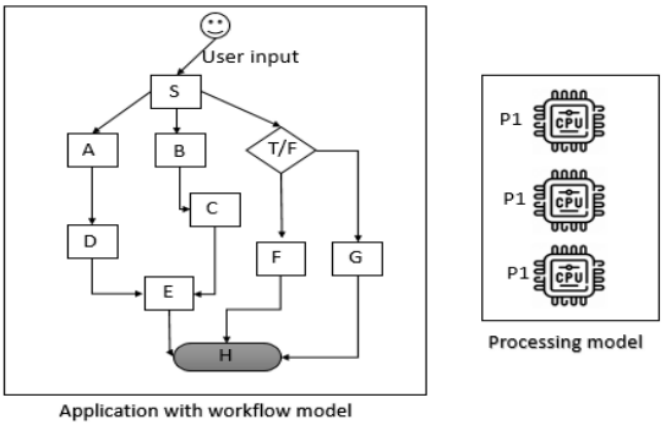

Figure 3. Workflow model, T: Task

An adaptive technique that assigns send an assignment to the processing in a manner that will cut down on the amount of time needed to create output that is needed to manage both of these challenges at the same time. An illustration of the problem with task scheduling seen in Figure 3

is shown in Figure 4. Processors are assigned a task via the scheduling mechanism. Figure 4 illustrates how this method distributes a job based on the processor's availability and priority. This method significantly reduces the amount of time that each activity takes to execute in parallel.

Goal and objective

To lower resource costs at the fog-cloud level, this research schedules every activity in the application pool based on processing unit availability. This study aims to:

a)Manage tasks at fog and cloud nodes based on their respective tasks. As cloud nodes offer processing power and storage, fog nodes benefit from bandwidth.

b) Arrange the tasks to utilize the processing unit best.

c) preserve task execution time and enhance system performance, create a fog broker task reassignment strategy for user-based deadline constraints.

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل للعلوم الصرفة والتطبيقية
ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

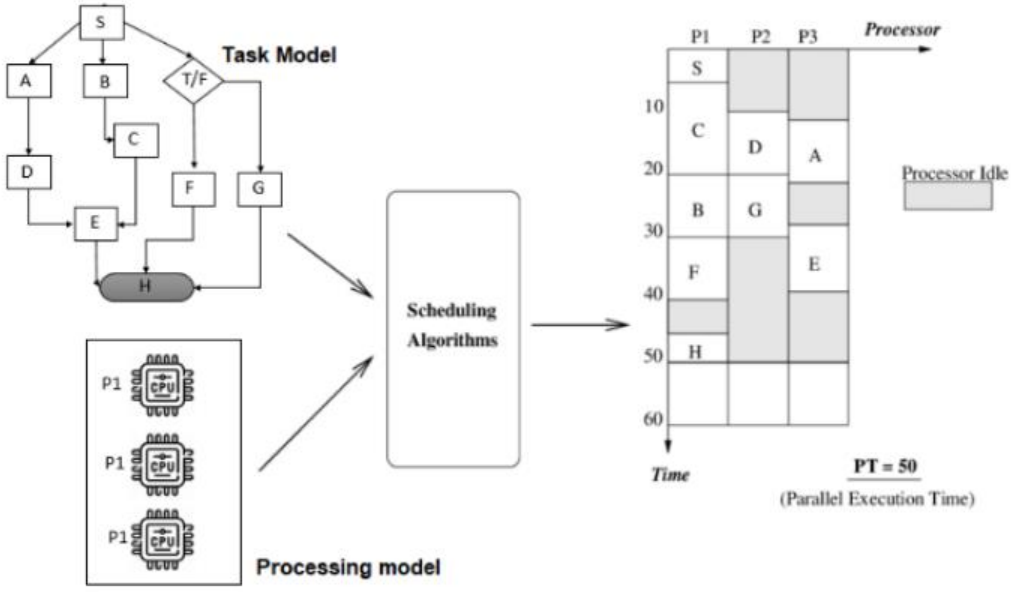

Figure 4. Task scheduling example [27]

Problem model

Task models and processor models make up the formulation of the job scheduling problem. To explain how the system is now implemented make assumptions about how things should be done when interacting with the application. The problem model is illustrated in Figure 5.

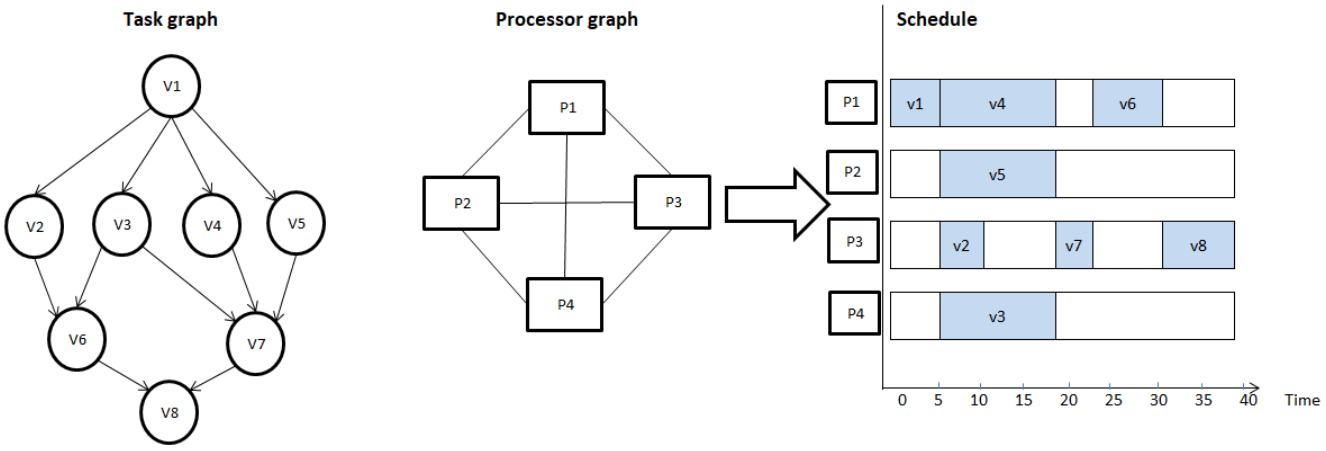

Figure5.illustrated of problem model[28]

METHOD

Three layers make up the architecture of fog computing [15]. Figure 6 illustrates this. IoT devices, fog computing, and cloud computing comprise the three levels. IoT devices make up the last layer. The layer in between is called the fog network. Fog computing is a middleman using cloud networks and Internet of Things devices. Fog computing uses the network's edge to give IoT devices quick access. It is positioned close to the application or end device. When using fog computing, all requests are routed by a fog node and fog servers that are situated locally at the network's edge rather than being sent to the cloud. The fog server functions as a cache for the user by locally storing frequently accessed data and retrieving it upon request. This allows the fog node to handle common and easy-to-solve issues.

Figure 6. Three-tier architecture of fog computing [29]

Task scheduling model

To efficiently execute fog computing task scheduling, the phases of task prioritization, the fog computing scheduling process includes node selection and task reassignment. The clouded computing job scheduling paradigm is shown in Figure 7. The suggested method generates and carries out a schedule according to the task graph's set of vertices and edges, as well as the processing pool's capacity. The timetable in place ensures that the total amount of time needed since the set of jobs is designed to minimize execution time and optimizes cloud computing resource management in terms of cost. The suggested task scheduling model consists of three primary components: job reassignment, node selection phase, and task prioritizing. Every task has a priority that is determined during the task prioritization phase. The task's orientation and placement within the process, as well as the data included in Task priority, is determined by application storage and data query.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

Task prioritization

Every task has a priority that is determined during the task prioritization phase. The priority of a job is determined by its direction and position within the workflow, as well as by the information found in data queries and application storage. This receives a score and is ranked higher. This might depend on how far the vertices are from the application's output. This examined the computation time at every vertex as well. Each node's priority in the task graph is determined by:

$$P(Vi) = \frac{wi}{\sum_{pn\epsilon N} Pn} + max\ [\frac{Cij}{\sum_{pn\epsilon N} bwn} + P(Vj)] \quad (1)$$

The location's priority value for the vertices is P(vi) in this equation. The computation time for that work is defined by Wi, while the communication delay between nodes I and J is denoted by C. Since the value of P(vj) for the first node is 0, the distance is also 0. On the other hand, this function computes that there are n nodes in each node, which is a priority value of n.

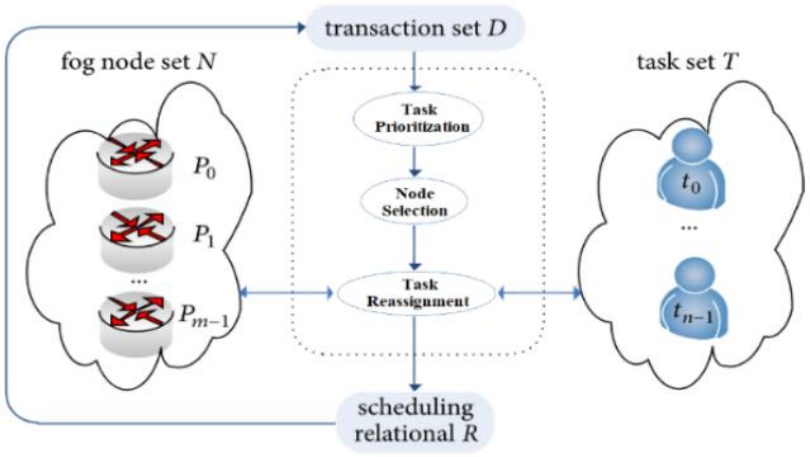

Figure 7. Fog computing layer

A phase of node selection

In the node selection step phase, a job is assigned to a node based on processor availability and priority. An assigned cloud node or fog level processing is used for the task if the required processing power is available at fog nodes. Each task in the pool only begins to run when all of its prior dependent job is finished. Data transfer time is the amount of time needed to finish all of its earlier dependent processes. The amount of time needed to send data between nodes is called communication time. The access time needed to gather all of the data from the application storage is the basis for the ready time. The task scheduler is ready to send jobs to the processor after analyzing this time. This phase's computed time is regarded as the ultimate optimum time needed

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

to complete each task found in the application pool. The entire work assignment schedule is the product of this phase.

The paper proposes an efficient approach for scheduling tasks and users algorithms, created especially for fog computing settings. It also contrasts the performance of this suggested algorithm with that of other scheduling algorithms that are currently in use, such as the Min-Min, Cost-Conscious, and Heterogeneous Early Complete-Time algorithms.

Pseudocode will cover the main steps for our scheduling algorithm:

Input:

- A list of tasks, each with computation requirements, delay constraints, and priority.
- A list of fog nodes, each with available resources and cost per unit of resource consumption.

Output:

- A mapping of tasks to fog nodes that minimizes the total cost.

Algorithm: Initialize total cost to 0

Create a list to store task-to-fog node mappings

Sort tasks by priority and delay constraints (higher priority tasks first)

Sort fog nodes by cost (lowest cost first)

For each task in the sorted list:

For each fog node in the sorted list:

If the fog node can accommodate the task (based on resources and delay):

Assign the task to the fog node

Update the fog node's available resources

Calculate the cost of assigning this task

Add the cost to the total cost

Add the task-to-fog node mapping to the list

Break the inner loop (move to the next task)

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

End If

End For

End For

Output the task-to-fog node mappings and total cost

Experimental analysis

For every scheduling strategy, a simulated environment and processing power with constant network connectivity are offered. The analysis's result is given as a quantity that allows for contrast. The main goal of the analysis is to compare the execution time, resources cost, or trade-off between cloud and fog computing. The developed greedy method for resource optimization for costs in a fog clouds scenario is compared. To maximize the resource cost, the reassignment scheme's accuracy is evaluated using a variety of deadline restrictions on the application that the user defines.

Analytical experimentation

The time complexity and resource cost of the suggested method are quantified to assess its performance. The suggested system was contrasted with different scheduling methods covered in the literature review. In this study, the simulation environment was used to evaluate performance. Because fog nodes are located close to Internet of Things devices, their bandwidth is higher than that of cloud nodes, despite their lower processing rate.

Evaluation of resource costs and execution time

To decrease performing time in IoT and cloud settings, fog nodes are introduced. When fog nodes are used, the only things needed for high computations to reduce task execution time are cloud services, proxy processing and fog-based cache memory. The resource consumption, communication time, and job completion time regardless of the fog computing layer are compared in this section. Figure 8 compares the times it takes to complete jobs, communicate with others, and use resources at fog clusters versus the cloud to highlight the suggested work contribution. A comparison between the fog computing layer's and the job execution time is shown in Figure 8(a). To assess the cost of resources, a simulation environment is provided. Four applications with varied instruction set lengths are examined under identical conditions, in addition to and instead of the clouded computing layer. By raising the cost element, this comparison seeks to raise the caliber of fog computing services. The normalized cloud cost value was utilized in this study; it was not employed in the research's section on cloud usage costs. For a fair and timely comparison,

مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل للعلوم الصرفة والتطبيقية
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

additional resource costs at fog nodes are taken into account in addition to cloud prices. A workflow-based application's resource costs are shown in Figure 8(b).

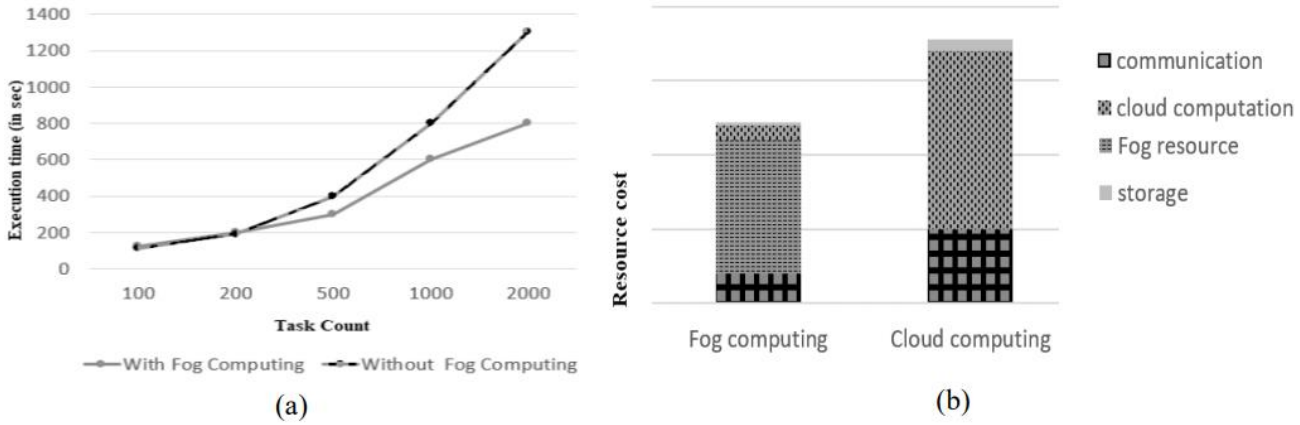

Figure 8 compares the amount of time needed to complete tasks, communicate with others, and use resources. It shows the differences in (a) execution time for both cloud and cloud computing and (b) the cost of resources.

In this comparison, the costs associated with storage, processing on cloud nodes, processing on fog nodes, and connectivity are taken into account. It has been noted that leveraging the locally available storage at the fog node offsets storage expenses in fog computing. The fog node's computational processing power allows it to process tiny jobs at the fog layer in addition to storage.

Scheduling method and the suggested algorithm

The time complexity and resource cost of the suggested method are quantified to assess its performance. Different scheduling strategies were compared to the suggested approach for comparison. Either a cost-conscious or a min algorithm was employed in this early multivariate complete-time method. Figure 9 illustrates the suggested contribution by contrasting the cost and execution time of the recommended algorithm with those of a full-time early approach and min-algorithm. The contrast between the suggested method's execution duration to that of the Figure 9(a) displays the min-min, cost-conscious, and heterogeneous early complete-time algorithms. Each of them maintains the same simulation environment, and performance is assessed for various applications with varying task counts. We examine the execution times of the min-min algorithm,

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

the diverse early complete-time technique and the cost-conscious algorithm for tasks with lengths of 30, 60, 90, and 120. It was found that in all four scenarios, the cost conscious algorithm takes the longest to execute. Conversely, the heterogeneous early complete-time algorithm has the shortest execution time. The suggested system consistently trails behind the heterogeneous method by 5 to 10%, even though the execution times there were many similarities between the diverse early complete-time method and the proposed approach. comparison of resource costs is just as significant as the comparison of execution times. The resource costs needed for the proposed method the comparison of the accurate, cost-aware, and heterogeneous full-time algorithms is shown in Figure 9(b). In applications with task lengths of 30, 60, 90, and 120, the resource costs of the min-min algorithm, A comparison is made between the economic algorithm and the initial complete-time algorithm, which is a heterogeneous, each of the four scenarios, it was found that the heterogeneous early completion algorithm had the highest cost resources. However, the algorithm that considers costs had the lowest cost of resources. The resource costs of the proposed algorithm are very close to those of the first complete-time algorithm that is diverse; nonetheless, the suggested system slows constantly 1% to 2% the heterogeneous algorithm's motivation.

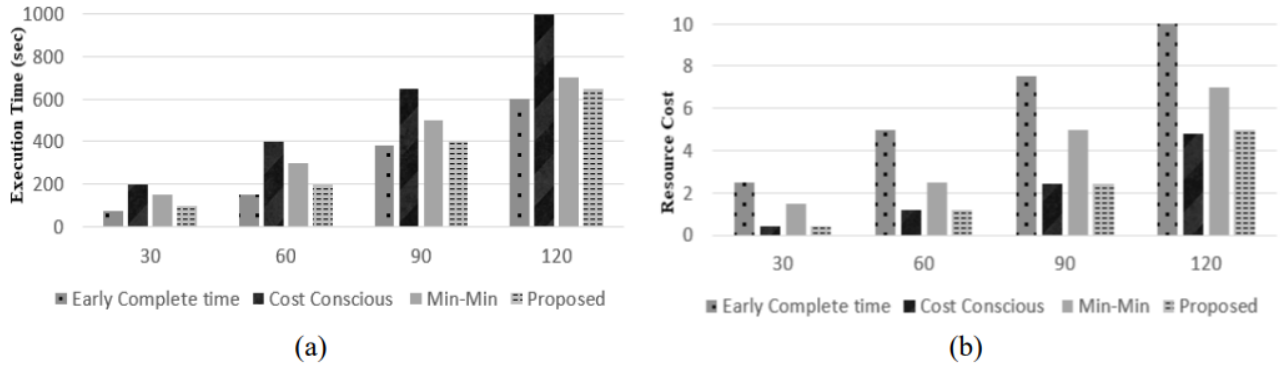

Figure 9 shows how the suggested method compares in comparison to a min-min algorithm, a cost-conscious method, and an early complete-time method in both in terms of cost and implementation time. Comparisons of resource costs and (a) execution times

Examination of deadline restrictions set by the user

The user sets limits and an advance date to guarantee the conclusion of the present application, thus the effectiveness analysis of the recommended system was contrasted with the suggested method without having to account for the data supplied by the user. The user sets deadline restrictions to preserve the caliber of services. Figure 10 shows the value of the work suggested by comparing the cost and execution time of the system algorithm while taking into consideration the user-specified deadline. A comparison of the suggested system's and algorithm's execution times without accounting for the user-specified deadline is shown in Figure 10(a). The algorithm that

performs better in the deadline constraint is shown in Figure 10(b).

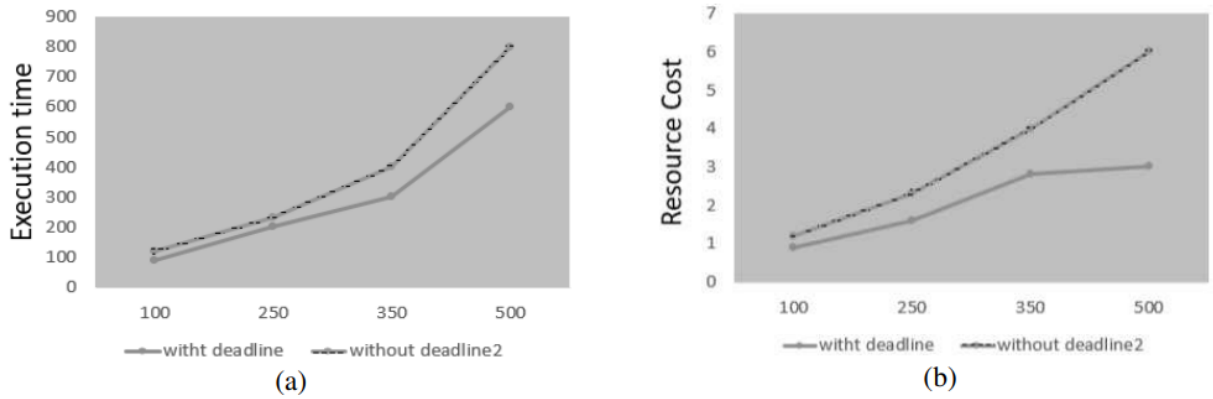

Figure 10. Comparison of the execution time of tasks and cost for the proposed system algorithm with and without considering the user-defined deadline, (a) Execution time comparison and (b) resource cost comparison

RESULTS AND DISCUSSION

By offering extremely users and task scheduling that is more economical while maintaining performance metrics, the suggested algorithm performs better than the current ones. It accomplishes this by making the fog node's concurrent tasks more efficient, which eventually reduces task delays and boosts system performance .To keep service quality high, user-defined deadline limits are implemented. The order of tasks is determined by various factors, including application storage, data query information, and task workflow. To ensure effective task allocation and completion, the priority of each task is set based on variables such as the distance between vertices the application output, and the computation time at each vertex.

CONCLUSION

The performance of several scheduling algorithms, such as the min-min, cost-conscious, and heterogeneous early complete-time algorithms, was constructed for comparison's sake. the early complete-time algorithm used a lot of resources. The resource cost for the Cost-Conscious method decreased significantly and was roughly equal to that of the suggested algorithm. Lastly, a comparison of the suggested system's re-assignment procedures is made.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

Conflict of interests.

There are non-conflicts of interest

References

[1] A. S. Abdalkafor, A. A. Jihad, E. T. J. I. J. O. E. E. Allawi, and C. Science, "A cloud computing scheduling and its evolutionary approaches," Indonesian Journal of Electrical Engineering and Computer Science, vol. 21, no. 1, pp. 489-496, 2021, doi: 10.11591/ijeecs. v21.i1.pp489-49.

[2] S. S. K. N. Prasad, "Quality and energy optimized scheduling technique for executing scientific workload in cloud computing environment," Indonesian Journal of Electrical Engineering and Computer Science, vol. 21, pp. 139-1047, 2021.

[3] M. Al-Khafajiy, T. Baker, A. Waraich, O. Alfandi, and A. Hussien, "Enabling high performance fog computing through Fog-2-fog coordination model," in 2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA), pp. 1-6,2019, doi: 10.1109/AICCSA47632.2019.9035353.

[4] Y. Y. F. P. M. F. Falah, S. Sukaridhoto, A. W. C. Tirie, M. C. Kriswantoro, B. D. Satria, and S. Usman, "Comparison of cloud computing providers for development of big data and internet of things application," Indonesian Journal of Electrical Engineering and Computer Science, vol. 22, pp. 1723-1730, 2021, doi: 10.11591/ijeecs.v22.i3.pp1723-1730.

[5] H. Hong, "From cloud computing to fog computing: unleash the power of edge and end devices," in 2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom),pp. 331-334,2017, doi: 10.1109/CloudCom.2017.53.

[6] S. N. B. Yerzhan, N. Seitkulov, G. B. Ulyukova, B. B. Yergaliyeva, and D. Satybaldina, "Methods for secure cloud processing of big data," Indonesian Journal of Electrical Engineering and Computer Science, vol. 22, pp. 1650-1658, 2021, doi: 10.11591/ijeecs.v22.i3.pp1650-1658.

[7] N. Bansal and A. K. Singh, "Effective task scheduling algorithm in cloud computing with quality of service alert bees and grey wolf optimization," Indonesian Journal of Electrical Engineering and Computer Science, vol. 25, pp. 550-560, 2022, do10.11591/ijeecs.v25.i1.pp550

[8] A. Rabay'a, E. Schleicher, and K. Graffi, "Fog computing with P2P: enhancing fog computing bandwidth for IoT scenarios," in 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData),pp82-89,2019.

[9] J. Garcia, E. Simó, X. Masip-Bruin, E. Marín-Tordera, and S. Sànchez-López, "Do we really need Cloud? Estimating the Fog computing capacities in the City of Barcelona," in 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion), pp. 290-295,2018, doi: 10.09/UCC-Companion.2018.00070.

[10] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fogcloud computing toward balanced delay and power consumption," IEEE Internet of Things Journal, vol. 3, no. 6, pp. 1171-1181, 2016, doi: 10.1109/JIOT.2016.25516.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

[11] B. Al-Otaibi, N. Al-Nabhn, and Y. Tian, "Privacy-preserving vehicular rogue node detection scheme for fog computing," (in eng), Sensors (Basel, Switzerland), vol. 19, no. 4, p.965, 2019, doi: 10.3390/s19040965.

[12] H. Al-Zoubi, "Efficient task scheduling for applications on clouds," in 2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/ 2019 5th IEEEInternational Conference on Edge Computing and Scalable Cloud (EdgeCom),pp. 10-13,2019, doi: 10.1109/CSCloud/EdgeCom.2019.00012.

[13] P. Phoc Hung and E.-N. Huh, "An adaptive procedure for task scheduling optimization in mobile cloud computing," Mathematical Problems in Engineering, vol. 2015, p. 969027, 2015/05/12 2015.

[14] G. Wang, Y. Wang, H. Liu, and H. Guo, "HSIP: A novel task scheduling algorithm for heterogeneous computing," Scientific Programming, vol.p. 76149, 2016/03/17 2016.

[15] L. Logeswaran, H. M. N. D. Bandara, and H. S. Bhathiya, "Performance, resource, and cost aware resource provisioning in the cloud," in 2016 IEEE 9th International Conference on Cloud Computing (CLOUD),pp. 913-916,2016, doi: 10.1109/CLOUD.2016.35.

[16] L. Liu, D. Qi, N. Zhou, and Y. Wu, "A task scheduling algorithm based on classification mining in fog computing environment," Wireless Communications and Mobile Computing, vol.p. 2102348, 2018, doi: 11155/2018/2102348.

[17] M. Abedi and M. Pourkiani, "Resource allocation in combined fog-cloud scenarios by usingartificial intelligence," in 2020 Fifth International Conference on Fog and Mobile Edge Computing (FMEC), pp. 218-222, 2020,doi: 10.1109/FMEC498.2020.9144693.

[18] H. Topcuoglu, S. Hariri, and W. Min-You, "Performance-effective and low-complexity task scheduling for heterogeneous computing," IEEE Transactions on Parallel and Distributed Systems, vol. 13, no. 3, pp. 260-274, 2002, doi:1109/71.993206.

[19] Y. Zhao, S. Cao, and L. Yan, "List scheduling algorithm based on pre-scheduling for heterogeneous computing," in 2019 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom), pp. 588-595 2019, doi: 10.1109/ISPA-BDCloud-SustainCom-SocialCom48970.2019.00089.

[20] N. Chopra and S. Singh, "Deadline and cost based workflow scheduling in hybrid cloud," in 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI),pp. 840-846,2013, doi: 10.1109.2013.6637285.

[21] V. Marbukh, "Towards fog network utility maximization (FoNUM) for managing fog computing resources," in 2019 IEEE International Conference on Fog Computing (ICFC), 2019, pp. 195-200, doi: 10.1109/ICFC.2019.00032.

[22] F. Bonomi, R. Milito, J. Zhu, and S. Adepalli, "Fog computing and its role in the internet of things," in Proceedings of the first edition of the MCC workshop on Mobile cloud computing, 2012, doi:10.1145/2342509.2342513.

[23] A. A. Alsaffar, H. P. Pham, C.-S. Hong, E.-N. Huh, and M. Aazam, "An architecture of IoT service delegation and resource allocation based on collaboration between fog and cloud computing," Mobile Information Systems, 2016, doi: 10.155/2016/6123234.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل للعلوم الصرفة والتطبيقية

[24] V. B. C. Souza, W. Ramírez, X. Masp-Bruin, E. Marín-Tordera, G. Ren, and G. Tashakor, "Handling service allocation in combined Fog-cloud scenarios," in 2016 IEEE International Conference on Communications (ICC),pp. 1-5,2016, doi:10.1109/ICC.2016.7511465.

[25] D. Zeng, L. Gu, S. Guo, Z. Cheng, and S. Yu, "Joint optimization of task scheduling and image placement in fog computing supported software-defined embedded system," IEEE Transactions on Computers, vol. 65, no. 12, pp. 3702-3712, 2016, doi: 10.1109/TC.2016.2536019.

[26] Y. Nan, W. Li, W. Bao, F. C. Delicato, P. F. Pires, and A. Y. Zomaya, "Cost-effective processing for Delay-sensitive applications in Cloud of Things systems," in 2016 IEEE 15thInternational Symposium on Network Computing and Applications (NCA), pp. 162-169, 2016, doi: 10.1109/NCA.2016.7778612.

[27] L. Huang and M. J. Oudshoorn, "Static scheduling of conditional parallel," Chinese Journal of Advanced Software Research, vol. 6, no. 2, pp. 121-12, 1999.

[28] W. Chen, G. Xie, R. Li, Y. Bai, C. Fan, and K. Li, "Efficient task scheduling for budget constrained parallel applications on heterogeneous cloud computing systems," Future Generation Computer Systems, vol. 74, pp. 1-11, 2017, doi: 10.116/j.future.2017.03.008.

[29] T. M. Hewa, A. Kalla, A. Nag, M. E. Ylianttila and M. Liyanage, "Blockchain for 5G and IoT: Opportunities and Challenges," 2020 IEEE Eighth International Conference on Communications and Networking (ComNet), Hammamet, Tunisia,pp.1-8, 2020,doi: 10.1109/ComNet47917.2020.930082.

مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل للعلوم الصرفة والتطبيقية
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

الخلاصة

يتكون هذا النظام من عقدة التحكم في الضباب وعقدة الوصول وتطبيق إنترنت الأشياء(IoT) , تهدف هذه الدراسة إلى تقليل تكاليف السحابة المرتبطة بالجدولة. يتم تنفيذ طريقة جدولة المستخدم والمهمة الاقتصادية لهذا السبب. تم إنشاء خوارزمية جدولة الوظائف لتحديد أفضل استراتيجية للتعامل مع العديد من الوظائف في عقدة وصول الضباب. تعمل تقنية إعادة التخصيص في النهاية على تقليل التأخير في كل من الوقت والخدمة. للأهداف التحليلية، تم إجراء عمليات محاكاة مكثفة وتمت مقارنة مقاييس الأداء بتلك الخاصة بالطرق الأخرى المستخدمة حاليًا. تم اكتشاف أن التقنية المقترحة تقلل من زمن انتقال المهمة وتوفر فعالية من حيث التكلفة من خلال تحسين القدرة المتزامنة في عقدة الضباب والمستخدمين وجدولة المهام مع إحصائيات الأداء المحسنة التي جعلت ذلك ممكنًا.

المواد والطرق:

يتناول البحث الحوسبة الضبابية، وهي بنية ثلاثية الطبقات تتكون من أجهزة إنترنت الأشياء، والحوسبة الضبابية، والحوسبة السحابية. تقع طبقة الحوسبة الضبابية بين شبكات السحابة وأجهزة إنترنت الأشياء(IoT) ، مما يوفر وصولاً سريعًا إلى هذه الأجهزة ومعالجة الحوسبة محليًا على حافة الشبكة.

النتائج:

يركز البحث على تقليل تكاليف السحابة في الجدولة. على الرغم من أن النظام المقترح أظهر العديد من أوجه التشابه في أوقات التنفيذ مقارنة بطريقة وقت الإكمال المبكر المتنوعة، إلا أنه يتخلف باستمرار عن الطريقة غير المتجانسة بنسبة 5 إلى 10٪.

الخلاصة:

يهدف هذا البحث إلى تقليل تكاليف جدولة السحابة لأجهزة إنترنت الأشياء (IoT). يتم تنفيذ خوارزمية جدولة تعتمد على المستخدم وتخصيص المهام الاقتصادي. يوصى باستراتيجية قائمة على القيود ومحددة من قبل المستخدم.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652