A Context-Aware Internet of Things (IoT) founded Approach to Scheming an Operative Priority-Based Scheduling Algorithms

Vandana Roy
Professor, GGITS, Jabalpur
Email: vandanaroy@ggits.org

Abstract

In recent years, smart computing has emerged as a promising and rapidly expanding field of technology. It senses the environment in real time and gives powerful analytics to perform intelligent decisions. Creating a scheduling algorithm based on priorities in order to decrease IoT process latency was the primary emphasis of the study challenge. The constraints of existing scheduling algorithms were investigated in order to build a scheduling algorithm that is based on priorities. We provide a context-based priority scheduling method to get around these restrictions. In order to determine which steps of the IoT process were crucial, we developed context attributes. Once the criticality has been identified, the proposed scheduling technique is used to schedule the IoT processes. The outcomes of the algorithms were confirmed using a variety of evaluation indicators. As demonstrated by the experimental results, the suggested scheduling algorithms outperformed the state-of-the-art techniques. Smart ATM uses a Case Study technique to analyse the algorithm. We identified the sensors that are part of the ATM and the settings in which they are relevant. We determined the priority value for each sensor. The processes are subsequently categorized according to their priority values. Then, a priority-based FCFS scheduling algorithm is used, and its performance is assessed using metrics like Average TAT, Cost, Energy, and the High critical process TAT ratio.

Keywords: IoT; TAT; EFCFS; PFCFS; Cost.

1. Introduction:

As smart computing has progressed, new technologies have been integrated to enable intelligent activities through the Internet of Things (IoT). However, due to the exponential increase in data generated by IoT devices, the cloud-connected devices have extremely long delays in receiving findings before they can be accessed by both the IoT devices and their users [1]. In order to complete a time-sensitive and mission-critical process, a quick choice must be made. The new paradigm of "edge computing," which seeks to run the process close to the data source, lends credence to this idea. The problem is that the Edge does not have enough resources to handle all of the smart environment's processes. In order to address this, we present a method that uses context and priority based scheduling to carry out the task that requires immediate attention [2]. The ultimate goal of a smart environment is to raise people's standard of living. In this proposed study, the smart computing environment is covered along with the Internet of Things. A new paradigm, the Internet of Things (IoT) uses sensor technologies built into IoT devices and cloud computing to make the world a smarter place [3]. According to several reports, the proliferation of IoT devices is causing an increase in data, which in turn causes smart environment services to have slower reaction times due to increased transmission latency imposed by the cloud server. With the advent of edge computing, which allows services to be given by means of the edge as a computing device, this transmission latency is being significantly decreased [4]. It is the context of the Smart Environment that determines the process's priority.

Grid Scheduling with Task Dependency: There are two distinct categories of work in a grid computing system: standalone jobs and collaborative endeavours. Figure 1 displays a task structure [5].

Doi : https://doi.org/10.54216/JCIM.130103
Received: May 11, 2023 Revised: August 17, 2023 Accepted: December 08, 2023

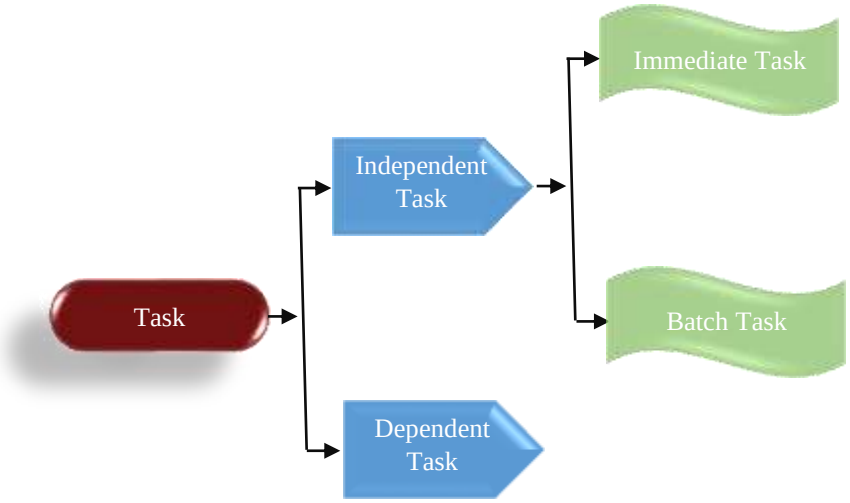

Figure 1: Task Hierarchy and Structure.

1.1. Independent task

Independent tasks are those that do not depend on the completion of any other work. That's why the computations won't be affected by the scheduling procedure. Individual responsibilities are represented in a matrix manner [6]. If two tasks Ti and Tj are truly separate, then there are two ways to schedule them: either Ti should come before Tj, or Tj should come before Ti. Meta tasks are another name for them. Both real-time and batch scheduling modes apply to the use of independent tasks.

1.2. Dependent Task

A dependent task is one that depends on the completion of another work. That's why it's important to consider scheduling when doing maths. Dependent tasks can be represented using either a Directed Acyclic Graph (DAC) or a Task Graph [7]. If two jobs Ti and Tj are interdependent, only the order in which they are performed, Ti then Tj, makes sense.

As the compute node reacts to the applications in both the fog and cloud environments, resource allocation becomes more complex [8]. The fog's computing nodes are dispersed throughout the network, while the cloud's are housed in a single, centralised facility. Because different applications have different needs in terms of processing power, data storage capacity, and network throughput, the IoT places a wide variety of demands on the computing nodes that must meet those needs [9].

Automated teller machine (ATM) transactions can be completed independently of bank tellers. The ATM Industry Association (ATMIA) estimated that there were more than 1.6 million ATMs in operation. Worldwide, there are around 1.6 million ATMs in operation, according to the ATM industry group. If you put an ATM in a commercial area, it will be safe from physical attacks like ram raids, cutting tools, plastic explosives, black box attacks, and skimming devices [10-12]. On the other hand, if you put an ATM on a highway, it will be vulnerable to all of the above.

The author has provided an overview of Diebold's Smart ATMs, which include a number of sensors linked to the cloud for remote monitoring. Issues with latency can arise in the event of physical attacks and fraud [13]. By applying the suggested technique, we may eliminate this delay and make the decision to activate the smart environment's actuator earlier, which in turn notifies the bank manager and ATM service provider [14]. Using a case study approach, one must follow these steps to evaluate an algorithm for context-based scheduling and priority in ATM systems:

1. It is necessary to determine the necessary contexts and anticipated priority values for every activity.
2. The numerical values of the context-specific coefficients are determined by implementing GLM within the Python statistics package.
3. The CPI is determined for every process, and the deterministic modelling method is used to catalogue the process's input data.
4. A process's criticality level is determined by the threshold value and then categorized as high, medium, or low.
5. A high priority queue buffer size of two is used in conjunction with the latency and expected execution time data to apply the proposed priority based PFCFS method.
6. The current FCFS scheduling method is used.
7. The final product is assessed using the HCP-TAT ratio, average TAT, energy consumption, and cost.

Doi : https://doi.org/10.54216/JCIM.130103
Received: May 11, 2023 Revised: August 17, 2023 Accepted: December 08, 2023

2. Related Work done:

Similar to the QoS-guided Min-Min algorithm, a heuristic approach called QoS Suffrage was presented. This method bases scheduling decisions on network bandwidth and suffrage value. This QoS Suffrage heuristic work scheduling algorithm is discovered to have a shorter makespan than the alternatives [15].

Suffrage heuristics provide the optimal solution by determining the time difference between the first and second minimums for completing a task. The resource that can complete the task in the least amount of time is given the higher-priority task [16]. Once a task is assigned, it is removed from the unmapped list and the process is repeated until all tasks have been mapped. This Suffrage algorithm has the same time complexity as the Max-Min heuristic. The approach improves grid performance and shortens makespan under balanced load. The authors make resource forecasts using data collected from previously completed jobs [17]. Researchers have presented a novel method that uses the heuristics of Max-Min as well as Min-Min. This heuristic algorithm makes its decision between the two aforementioned techniques by considering the variance in the estimated times at which each work may be completed on each machine [18].

The architects of the Grid-JQA architecture developed a set of scheduling methods based on quality of service. This algorithm utilises matrices to pair n jobs with m available assets. Although this heuristic technique outperforms Max-min, Min-min, and Suffrage, it is not a realistic approach and is seen as unrealistic for computational issues. Workflow execution time reduction is proposed. This algorithm treats a grid system like a queue, with jobs being distributed across the members of the queue [19]. This approach is system oriented and takes into account the execution cost of activities. It is demonstrated to be useful for monetary grids. The time this algorithm takes is hardly appreciable unless the problem's nature changes drastically [20].

The authors have developed a user-oriented grid task scheduling algorithm that takes into account advanced reservation and resource selection. The goal of this task scheduling algorithm is to minimise the total running time of the individual tasks without considering the total running time of the submitted jobs [21]. The overall makespan of the system is not reduced, as demonstrated by the results of this algorithm.

The Multiple Resources Scheduling (MRS) method was developed to evenly distribute work in grid systems. This method takes into account the site's capabilities and resource requirements [22]. According to the collected results, this method is preferable to traditional backfill and replication algorithms in terms of average waiting time of tasks, completion time of queue, and mean resource utilisation.

In order to fairly divide up jobs across the grid's resources, the authors described a novel algorithm. The overall completion time of jobs on the grid cannot be decreased, despite the fact that this approach balances load and enhances throughput. It was proposed to use a priority-based task scheduling algorithm (P-TSA) in a grid environment [23]. This method use a priority ordering to determine which jobs should be completed first. When making plans, prioritise the resource whose completion time is shortest. When compared to the Min-Min and Max-Min Grid Task Scheduling Algorithm, this one produces superior results. Scheduling jobs in a grid using the Semi-Interquartile Min-Min Max-Min (SIM2) technique optimises makespan and resource utilisation while decreasing task completion times.

Table 1: Comparison of Real-Time Stream Data Processing Methods.

Method	Latency (milliseconds)	Accuracy (percentage)	Scalability	Resource Utilization (percentage)	Data Integrity (percentage)	Security and Privacy (percentage)
Sliding Window Approach	50	85	High	70	90	80
Online Machine Learning	70	78	Medium	60	85	75
Data Stream Clustering	60	80	High	65	88	70
Stream Data Compression	40	90	Medium	75	92	85
Incremental Statistical Analysis	55	87	High	80	89	82
Distributed Stream Processing	30	92	High	85	95	88
Real-Time Database Systems	35	91	High	90	93	90
Complex Event Processing	65	82	Medium	70	87	77

Doi : https://doi.org/10.54216/JCIM.130103
Received: May 11, 2023 Revised: August 17, 2023 Accepted: December 08, 2023

Online Time Series Analysis	58	86	High	75	91	80
Dynamic Data Visualization	80	75	Low	50	84	70

Table 1 evaluates ten real-time data processing methods which is based on key performance parameters. It provides insights into their capabilities and limitations, aiding researchers in method selection for various applications.

The increased load imbalance on the computing grid, a disadvantage of the MET heuristic, is mitigated. He used a two-stage heuristic that he put into effect. In the first stage, tasks are given to the resource with the shortest runtime. The second step involves redistributing workloads from overburdened to underutilised resources [24].

Scheduling jobs that don't depend on one another is done with QoS priority based scheduling. There was a novel advancement in batch mode algorithms with the introduction of the Longest Job to Fastest Resource - Shortest Job to Fastest Resource heuristic. This heuristic, known as LJFR-SJFR (Longest Job to Fastest Resource-Shortest Job to Fastest Resource), begins with a collection of all unmapped tasks. The quickest possible way to finish a task is determined, much like MinMin. Shortest Job to Fastest Resource is the default setting. The longest job is then assigned to the resource with the quickest turnaround time. He also describes the Work Queue Heuristic algorithm for autonomously scheduling tasks. This algorithm distributes jobs to computers at random.

The author modified the Fog Computing infrastructure to include a new method of scheduling tasks. The authors studied load balancing, which manages data consistency with minimal complexity and ensures that actual tasks are completed on time.

For scheduling tasks, researchers suggested basing an algorithm in the fog layer on priority tiers. The algorithm is important and effective since it prioritises the preparation of the jobs with the highest due dates first.

In order to speed up the process of creating sets of frequently used items, the author proposed a I-Apriori algorithm by tweaking the original Apriori algorithm. Similar to the I-Apriori technique, a new algorithm for task scheduling in fog computing (TSFC) is proposed. In the TSFC work scheduling algorithm, the association rules it creates are crucial. The fog node with the earliest completion time is assigned the work with the shortest end time.

3. The Proposed Work:

Here we present a tabular comparison of the results of a deterministic modelling comparison between the proposed algorithm and the current algorithm in the ATM system. For the purpose of testing the suggested method, the sensors specified in are utilized. Security alerts, ATM diagnostics, and environmental sensing are all made possible by the sensors. All of the sensors used in experiment 0 are based on the author's suggested context models; these include sensors for high- and medium-critical processes, as well as a smart meter sensor that operates in a low-critical context. It is the authors' low-critical process that periodically communicates with the cloud servers.

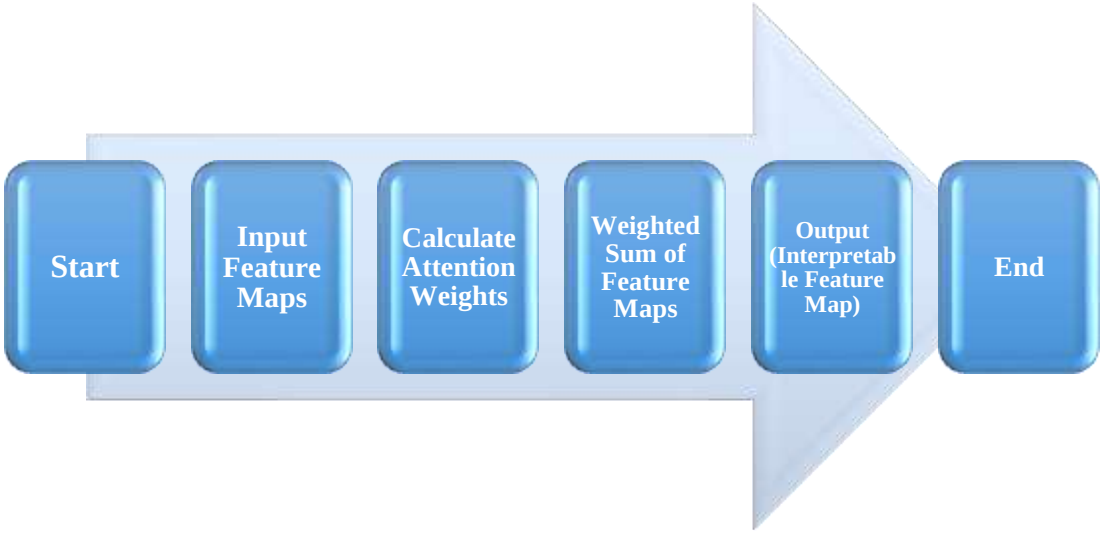

Figure 2: Flowchart for Attention Mechanism.

The following steps need to be taken in order to evaluate an algorithm for context-based scheduling and priority in ATM systems. This evaluation must be done using a case study approach.

3.1. The ATM's sensor has its context data set.

3.2. Obtain the value of the coefficients for each context.

Doi : https://doi.org/10.54216/JCIM.130103
Received: May 11, 2023 Revised: August 17, 2023 Accepted: December 08, 2023

The influence of each context in the IoT process is evaluated to determine the qualified priority value, which is Ŷ, which is based on the context range value within the criticality level alone. To do this, we use GLM on the Ci context values and the Y projected priority values. Every situation affects the priority value e. The values of each contex's coefficients decide this.

This adversarial training process continues iteratively until a balance is reached, where the generator produces threat scenarios that are highly realistic. These generated scenarios can then be used for cybersecurity threat forecasting, providing valuable data for assessing potential threats and vulnerabilities.

$$LG=-\mathrm{E}z[\log D(G(z))] \quad (1)$$

The discriminator's goal is to reduce:

$$LD=-\mathrm{E}x[\log D(x)]-\mathrm{E}z[\log(1-D(G(z)))] \quad (2)$$

3.3. Next, we will calculate the CPI.

The CPI value (real priority value) of each process is determined by the calculated coefficient, and operations are categorized according to this value. Given a dataset D with features X and retention labels Y, the decision tree is recursively split based on feature Xj and threshold T to maximize information gain IG. The splitting criterion is defined as:

$$IG(Xj,T)=H(D)-v\in\text{values}(Xj)\sum|Dv|/|D|H(Dv) \quad (3)$$

3.4. Sort the procedure

Each process is categorized into high, medium, and low critical processes after determining its CPI. Based on the threshold value, the processes are categorized into three tiers to establish their priority. The network consists of an input layer X, multiple hidden layers Hi with activation functions σ, and an output layer Y representing retention probabilities. The forward pass equations for each layer i are given by:

$$Zi=XWi+Bi \quad (4)$$

$$Hi=\sigma(Zi \quad (5)$$

$$Y=\sigma(Hn-1Wn+Bn) \quad (6)$$

3.5. Use the current FCFS scheduling technique.

Instead of categorizing processes, the current scheduling method uses FCFS to only schedule them. It is necessary to establish the Edge's high priority queue. A value of two is assigned to the buffer size of the high priority queue. To schedule the remaining processes in the cloud, we use the current FCFS Scheduling algorithm in the EDGE_CLOUD Architecture. The first two processes are scheduled at the edge.

This algorithm is essential for preventative threat mitigation since it tells businesses the kinds of risks they may expect to face soon.

$$St+1=LSTM(St,St-1,...,St-n) \quad (7)$$

where S_t+1 is the anticipated danger scenario and n is the number of items in the sequence.

3.6. Implement the Suggested Scheduling Method.

The suggested priority-based FCFS scheduling algorithm uses the edge to execute processes in the high-priority queue; the remaining processes are either run in the cloud or, if the edge is unavailable, in the edge. When the high priority queue is not empty, processes are scheduled at the edge because the size is defined as two. If the high priority queue is empty and the Edge resource is available, any process with medium or low priority will be scheduled at the edge.

3.7. Assess the Suggested Scheduling Method

By contrasting the outcomes with the current FCFS scheduling algorithm, the suggested priority-based approach is assessed. Calculated variables include average TAT, HCP_TAT ratio, process energy consumption, and total process cost.

4. Result and Discussion:

But this statistic adds greater weight to the evaluation of the scheduling algorithm, according to the literature review. To measure performance on criteria like, we simulate a thorough comparison of new scheduling techniques with current scheduling algorithms.

1, Average TAT:

What we call "turn around time" is the amount of time it takes to go from submitting a process to having it executed and finally finished. It includes waiting time, processing time, and latency, which is the time it takes for data to reach the processing device. Since TAT includes transmission time, waiting time, and process execution time, it is used for evaluation.

2. Cost:

The execution cost and transmission cost make up the total cost of an IoT process. Bandwidth utilized for transmission is part of the transmission cost, while CPU use for an IoT process is part of the execution cost. At the CLOUD and EDGE devices, cost is measured in $/process.

3. Energy:

Both the computation energy and the transmission energy are part of the total energy required by an operation. A unit of power, Watts (W), is used to measure cost.

Doi : https://doi.org/10.54216/JCIM.130103
Received: May 11, 2023 Revised: August 17, 2023 Accepted: December 08, 2023

4. HCP_TAT ratio

Prioritizing mission-critical tasks is at the heart of the priority-based scheduling method that has been presented. In order to demonstrate the extent to which the new algorithm outperforms the existing method, this metric zeroes in on the TAT of the high critical process in both the proposed and existing algorithms. The HCP_TAT ratio is the comparison between the current algorithm's maximum TAT for the high critical process and the proposed algorithm's corresponding TAT.

Another statistic that shows how efficient the proposed algorithm is the HCP TAT ratio, which compares the current scheduling algorithm with the proposed algorithm by looking at the time it takes to complete high-critical processes. The HCP_TAT ratio is the comparison between the current algorithm's maximum TAT for the high critical process and the proposed algorithm's corresponding TAT.

Table 2: Average TAT comparison of High Critical Process.

S. No.	High Priority Process	Existing FCFS TAT	Proposed Priority FCFS TAT
1	P1	1.29	1.31
2	P3	4.25	1.86
3	P8	65.78	2.89
4	P9	69.26	4.45
5	P14	133.94	3.36
6	P15	137.67	7.28

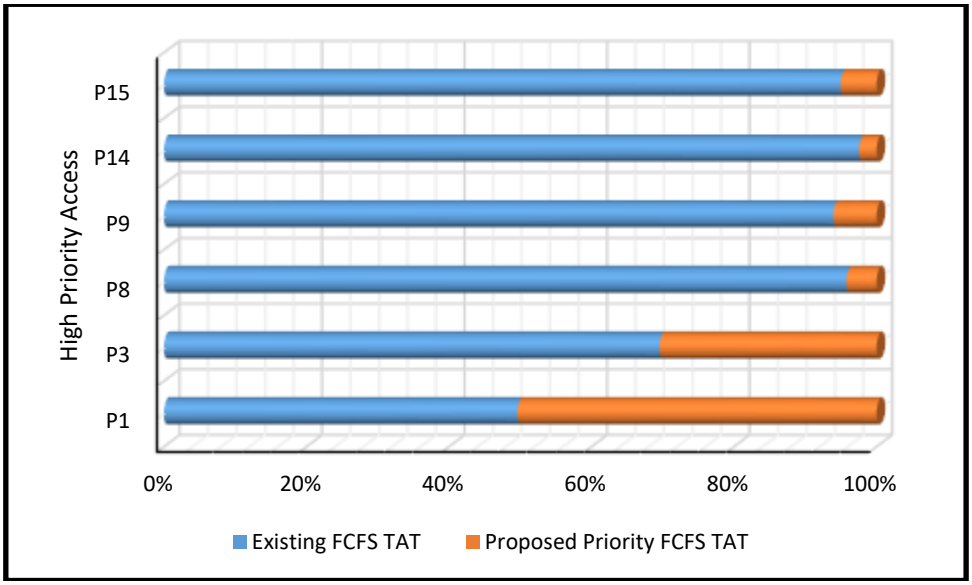

Figure 3: Average TAT comparison of High Critical Process.

The current and scheduling algorithms' TATs for high-critical processes are shown in Table 2. To reduce latency and TAT for high-critical processes, the proposed priority-based FCFS algorithm outperforms the current FCFS method by a margin of 94.76% when it comes to running these processes at the edge.

Table 3: Evaluation parameters comparison of the proposed method.

S. No.	Evaluation Parameters	Algorithms	
		EFCFS	PFCFS
1	Average TAT(ms)	69.75	21.82
2	Energy (W)	9.36	2.08
3	Cost($)	48.59	23.28
4	HCP_TAT ratio(ms)	312.87	2.59

Doi : https://doi.org/10.54216/JCIM.130103
Received: May 11, 2023 Revised: August 17, 2023 Accepted: December 08, 2023

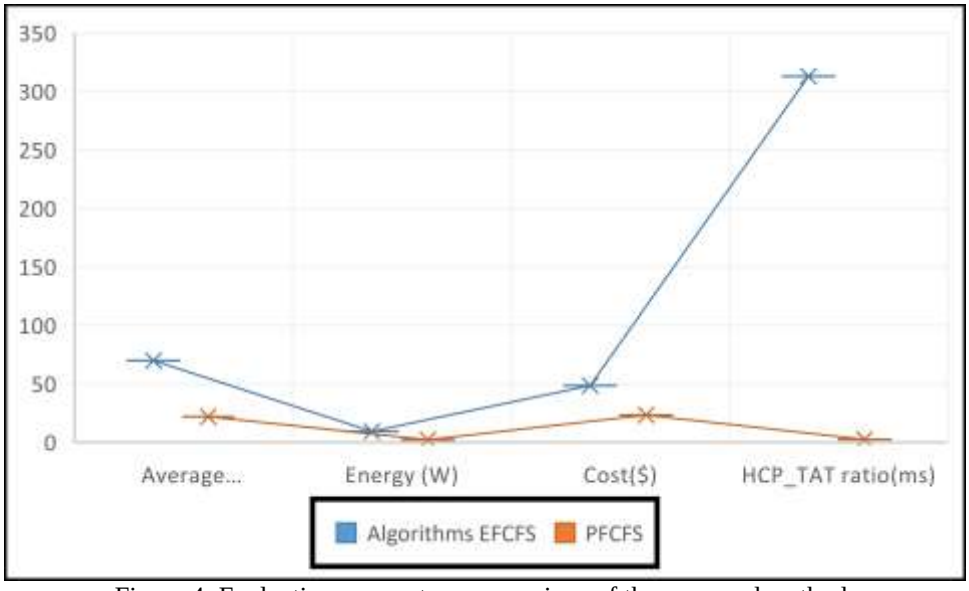

Figure 4: Evaluation parameters comparison of the proposed method.

It is clear from Table 3 that the average TAT, cost, and energy values all rise with an increase in the number of processes. Table 3 shows that relative to the current FCFS method, the suggested scheduling approach consistently has lower high priority process latency. In a smart environment, processes are scheduled in response to events based on a variety of criteria, including dependencies, reliability, activity, security, availability, and user-defined contexts. As a result, processes with higher priority are executed earlier. Process scheduling can be done anytime the Edge is available with minimal latency using the suggested P-FCFS method, as shown in Table 3.

4. Conclusion and Future Scope:

Recently, smart computing has emerged as a promising and rapidly expanding field of technology. It senses the environment in real time and gives powerful analytics to perform intelligent decisions. Cloud computing, machine learning, sensor-based technologies, the internet of things (IoT), and edge computing are all components of smart computing. The Internet of Things (IoT) is one such technology; it facilitates connection between computers and millions of other devices, which in turn makes many people's lives easier and more efficient. When these conversations take place in the cloud, the results take a long time to arrive. So, to get around the latency problems, Edge Computing came into play. By moving processing to the edge rather than the cloud, we can cut down on the latency of IoT processes. On the other hand, due to resource constraints, the Edge cannot handle all processes. Consequently, a method is suggested for lowering the Smart Environment's IoT process latency. In this article, we will take a look at the ATM system and the ways it can be attacked. Also, other research papers provided a list of the sensors used by ATM systems. Using both the current FCFS scheduling method and the new priority based FCFS algorithm, the case study on ATM systems was conducted. An evaluation was conducted on the algorithm's output. When compared to the current scheduling algorithm, the findings show that the suggested method is superior in terms of energy consumption, cost, average TAT, and HCP_TAT ratio. The algorithm that has been suggested does not take preventative measures. The method has potential for use in pre-emptive scheduling algorithms down the road.

References

[1] S. Agarwal, S. Yadav, and A. K. Yadav, An Efficient Architecture and Algorithm for Resource Provisioning in Fog Computing, International Journal of Information Engineering and Electronic Business, 8 (1) (2016), 48-61.

[2] L. Ni, J. Zhang,C. Jiang,C. Yan, and K. Yu, Resource Allocation Strategy in Fog Computing Based on Priced Timed Petri Nets, IEEE Internet of Things Journal, 4 (5) (2017), 1216–1228.

[3] X. Xu, S. Fu and Q. Cai, Dynamic Resource Allocation for Load Balancing in Fog Environment, Wireless Communications and Mobile Computing 2018, Article ID 6421607 (2018), 15 pages.

[4] S. Bitam, S. Zeadally, and A. Mellouk, Fog Computing Job Scheduling Optimization Based on Bees Swarm, Enterprise Information Systems, 12 (4), (2017), 373-397.

[5] H. J. Hong, From Cloud Computing to Fog Computing: Unleash the Power of Edge and End Devices. Proceedings of IEEE International Conference on Cloud Computing Technology and Science (CloudCom) (2017), 331-334.

[6] R. K. Naha, S. Garg, D. Georgakopoulos, P. P. Jayaraman, L. Gao, Y. Xi-ang, and R. Ranjan, Fog Computing: Survey of Trends, Architectures,Requirements, and Research Directions, IEEE Access, (2018).

Doi : https://doi.org/10.54216/JCIM.130103
Received: May 11, 2023 Revised: August 17, 2023 Accepted: December 08, 2023

[7] R. Vijayalakshmi and V.Vasudevan, Scheduling Independent Obligations on Heterogeneous Computing Systems by Heuristic Approach, International Journal of Pure and Applied Mathematics, Special Issue, 119 (12) (2018), 13983-13989, ISSN: 1314-3395 (on-line version), url: http://www.ijpam.eu.
[8] Yang, S.; Tian, K.; Liu, R. Task scheduling algorithm based on value optimisation for anti-missile phased array radar. IET Radar Sonar Navig. 2019, 13, 1883–1889.
[9] Liu, Y.; Lu, X.; Zhang, Y.; Mou, Z. A task scheduling algorithm for phased array radar based on dynamic time window. In Proceedings of the IET International Radar Conference (IET IRC 2020), Online, 4–6 November 2020.
[10] Liu, D.; Zhao, Y.; Cai, X.; Xu, B.; Qiu, T. Adaptive scheduling algorithm based on CPI and impact of tasks for multifunction radar. IEEE Sens. J. 2019, 19, 11205–11212.
[11] Huang, L.; Zhang, Y.; Li, Q.; Pan, C.; Song, J. Task-scheduling scheme based on greedy algorithm in integrated radar and communication systems. J. Eng. 2019, 2019, 5864–5867.
[12] Zhu, X.; Yang, R.; Li, X.; Yuan, K. Multifunctional integrated system resource scheduling method based on improved GA-PSO. J. Air Force Early Warn. Acad. 2021, 35, 202–206.
[13] Huang, Y.; Hu, S.; Ma, S.; Liu, Z.; Xiao, M. Designing low-PAPR waveform for OFDM-based RadCom systems. IEEE Trans. Wirel. Commun. 2022, 21, 6979–6993.
[14] Li, W.; Ren, P.; Xiang, Z. Waveform design for dual-function radar-communication system with golay block coding. IEEE Access 2019, 7, 184053–184062.
[15] Rong, J.; Liu, F.; Miao, Y. High-efficiency optimization algorithm of PMEPR for OFDM integrated radar and communication waveform based on conjugate gradient. Remote Sens. 2022, 14, 1715.
[16] Rong, J.; Liu, F.; Miao, Y. Integrated radar and communications waveform design based on multi-symbol OFDM. Remote Sens. 2022, 14, 4705.
[17] Rong, J.; Liu, F.; Miao, Y. Multifunctional waveform design method combining AC-ICF and TR techniques. J. Signal Process. 2020, 36, 1721–1726.
[18] Li, C.; Bao, W.; Xu, L.; Zhang, H.; Huang, Z. Radar communication integrated waveform design based on OFDM and circular shift sequence. Math. Probl. Eng. 2017, 2017, 9840172.
[19] Lin, Z.; Lin, M.; Wang, J.; Cola, T.; Wang, J. Joint beamforming and power allocation for satellite-terrestrial integrated networks with non-orthogonal multiple access. IEEE J. Sel. Top. Signal Process. 2019, 13, 657–670.
[20] Lin, Z.; An, K.; Niu, H.; Hu, Y.; Chatzinotas, S.; Zheng, G.; Wang, J. SLNR-based secure energy efficient beamforming in multibeam satellite systems. IEEE Trans. Aerosp. Electron. Syst. 2022.
[21] Khan, M.J.; Khan, M.A.; Malik, S.; Kulkarni, P.; Alkaabi, N.; Ullah, O.; El-Sayed, H.; Ahmed, A.; Turaev, S. Advancing C-V2X for Level 5 Autonomous Driving from the Perspective of 3GPP Standards. Sensors 2023, 23, 2261.
[22] Pradhan, R.; Satapathy, S. Energy-Aware Cloud Task Scheduling algorithm in heterogeneous multi-cloud environment. *Intell. Decis. Technol.* 2022, *16*, 279–284.
[23] Chen, H.; Liu, G.; Yin, S.; Liu, X.; Qiu, D. ERECT: Energy-Efficient Reactive Scheduling for Real-Time Tasks in Heterogeneous Virtualized Clouds. *J. Comput. Sci.* 2017, *28*, 416–425.
[24] Duan, H.; Chen, C.; Min, G.; Wu, Y. Energy-aware scheduling of virtual machines in heterogeneous cloud computing systems. *Future Gener. Comput. Syst.* 2017, *74*, 142–150.

Doi : https://doi.org/10.54216/JCIM.130103
Received: May 11, 2023 Revised: August 17, 2023 Accepted: December 08, 2023