A Deadline-Aware Priority based Semi Greedy Task Scheduling Technique in Fog Computing

Shipra Gautam[1, *, †] and Amarjit Malhotra[2, †]

[1,2] *Department of Information Technology, Netaji Subhas University of Technology, New Delhi, India, 110078*

Abstract

In the recent past, swift evolution of Internet of Things devices fabricates a diverse range of real-time applications which necessitate low latency and real-time response. Day by day there is a rise in the total number of IoT devices that induce high traffic and connection delay in network that connects cloud with end devices. Fog computing solves certain problems by bringing cloud computing near to the end devices which results in better service quality for requested tasks. It appears in the middle of the cloud layer and Internet of Things (IoT) users. The key benefits of fog computing are efficient usage of resources and decreasing latency for end users. Owing to the limited accessibility of resources in fog environment the most substantial challenge of fog computing is optimum assignment of the tasks to fog nodes (FN). Although, due to complex and firm quality of service requirement, assigning resources to tasks is rigorous. A task scheduling algorithm is efficient if it reduces the usage of energy and performs tasks within their deadline. Here, we formulate a novel strategy for scheduling task and managing resources to meet deadlines, minimizing energy consumption and optimizing makespan. We propose a heuristic algorithm to address task scheduling issue using priority based semi greedy strategy (PSGS). The primary objective of proposed strategy is to enhance the system's overall energy efficiency while still adhering to deadline of the task. This approach tracks the severity level of the task by considering its deadline and arrival time. The performance of this strategy is tested and results confirm that the proposed PSGS strategy enhances the deadline satisfied tasks by 12.6% with respect to second best baseline algorithm and 23.9% with respect to detour. Also, the PSGS reduces energy consumed by the fog devices and achieves optimal makespan.

Keywords

Fog Computing, Energy consumption, Task scheduling, Semi-greedy strategy, Internet of Things, Cloud Computing, Deadline, Resource management[1]

Symposium on Computing & Intelligent Systems (SCI), *May 10, 2024, New Delhi, INDIA*

* Corresponding author.
† These authors contributed equally.
tauras.shipra@gmail.com (S. Gautam); amarjit.malhotra@nsut.ac.in (A. Malhotra)
0009-0009-1577-0175 (S. Gautam)

1. Introduction

Cloud computing is an emerging, rapidly advancing and widely known new distributed computing in the era of the internet. It has a huge economic benefit in most of the fields. As a result of proliferation of intelligent devices, AI & other Internet of Things [1], the connection between the objects has become stronger and there is an explosive growth in the number of devices that have been connected with the network. So, the edge devices need high reliability and low latency. The primary goal of the Internet of Things is to assess, process and handle data generated by numerous gadgets in the cloud. This involves utilizing cloud computing, where the substantial volume of data collected by interconnected devices, although individually small, can be efficiently processed [2,3]. There is an ultra-long distance between the cloud data center and the end users which diminishes the real-time performance. The substantial volume data that has been collected from the processing of IoT applications has been carried out in the cloud data center, leading to an augmentation [4]. In order to address the constraints of current technology related to latency, mobility support, and improve user satisfaction, Cisco coined a term Fog Computing in 2014 [5,6].

Cloud computing data centers are situated at a distance from customer convenience, leading to extra network communication expenses and latency when handling extensive data [7,8]. Because of this constraint, cloud computing cannot effectively support the services of millions of customers distributed globally. Due to the rigorous latency requirement the time and the preference of the resources being allocated is crucial. Fog and cloud computing technologies deliver services to users as needed. These arising technologies are viewed as the foundation for all Internet of Everything (IoE) applications. However, cloud computing is not only a viable choice for delay-tolerant applications but demands real-time responses.

Fog computing expands the scope of cloud computing, extending it from the central core to the network's edge providing all the services like computing, storage & networking service in between traditional cloud and edge devices which leads to reduction in latency and network congestion [9,10,11]. The basic architecture of fog computing is mainly constructed of three separate operational layers, specifically terminal layer, fog layer and third one is cloud. Previously, all the data has been transferred to the cloud environment for further examining and analysis that leads to wastage of network bandwidth and resources. Therefore, to address these challenges fog computing becomes relevant. In particular, fog computing is needed when a massive amount of data has been produced from real time applications that require low latency computing. Unlike cloud computing, in fog computing resource allocation is the structured approach to allocate available resources to the requested users over the internet.

In the presented research, a resource management technique is illustrated to prioritize the tasks grounded on their potency. The tasks are arranged in the priority queue to allocate resources. This research has taken forward to elevate effectiveness of managing fog resources via scheduling of tasks. The proposed approach is contrasted with the existing baseline algorithms and the execution shows that PSGS is better from baseline algorithms.

The remainder of the study is arranged as follows: Section 2 illustrates recent work in the field of scheduling tasks in foggy environment. Section 3 elaborates the detailed explanation of the system model, formulation of the problem and its working. Section 4 represents the proposed algorithm. Section 5 represents the simulation and result analysis of the addressed algorithm. Finally, the conclusions with future challenges are discussed.

2. Related Work

Azizi, et al. [12] considered the task scheduling constraint along with the intention of scaling down energy consumed by fog resources during meeting the deadline of requested tasks. Author proposed two techniques specifically Priority-aware Semi-Greedy and the PSG-M method seeks minimizing deadline violation time. The proposed strategy aims to raise the probability of tasks that fulfill their deadline, & optimizes makespan and the energy usage.

Najafizadeh et al. [13] presented a Multi-Objective Simulated Annealing. This strategy uses deadline as a basis for secure tasks distribution on cloud and fog. To assess the effectiveness of the suggested approach, three algorithms, including MOPSO, MOTS, and MOMF are considered. It has accomplished much better performance with reference to access level control, delay time of service, deadline and satisfactory results in terms of service cost.

Yadav, et al. [14] presented a modified fireworks algorithm that combines opposition-based learning with differential evolution technology to lower the cost, makespan and enhance resource utilization. Researcher also explore new issues on multi-objective scheduling problems, contemplate extra optimization goals, such as execution makespan and energy-saving. The QoS and financial cost are significantly influenced.

Kaur et al. [15] proposed Task-Resource Adaptive Pairing for efficient scheduling which concurrently lowers the delay, cost and energy usage. Author investigates task-resource optimization for efficient scheduling. This technique is also organized as a multi-purpose optimization issue for arranging delay-tolerant task in fog computing.

Wang et al. [16] proposes a technique I-FASC for allocating resources using (I-FA) enhanced genetic algorithm as there is increasing needs for IoE applications. IFA introduces the explosion radius detection mechanism of fireworks. An improved firework algorithm enhances the load balancing and reduces processing time for tasks.

Fizza et al. [17] considers real-time capabilities of tasks during scheduling in the fog environment. For scheduling individual tasks on a designated processor, the EDF algorithm is used. The authors categorize tasks into three types mainly hard, firm and soft. After the scheduled conditions are satisfied then they aim to arrange hard or challenging tasks to embedded devices, firm tasks on fog processors and soft tasks on cloud devices which minimizes total communication delay.

Abdel-Basset et al. [18] come up with a marine predators-based task offloading framework that improves QoS in the fog-cloud domain. To address the delay-sensitive and task scheduling with a focus on energy-efficiency in IoT edge computing a heuristic algorithm was proposed. The authors introduced two iterations of the proposed model. The first one is Modified MPA, which enhances the exploitation capability of the MPA by incorporating the most recently recorded positions rather than the last outstanding one. The next iteration is the Improved MMPA method, which undergoes further enhancement through a reinitialization and mutation process based on a ranking strategy toward the best approach.

Ghanavati et al. [19] introduced AMO that stands for Ant Mating Optimization and it was applied to address the task allocation challenge in the fog computing domain. Author suggested a model to address the rising need for computational resources and to opt the most effective process for assigning tasks to fog nodes and the result shows that this approach has less energy utilization.

Binh et al. [20] proposed TCaS framework for time and cost-aware scheduling. The capability of addressed strategy is evaluated with regards to several task-based data-set on cloud-fog devices. A balance between processing time, cost and user's Contentment was established through optimization criteria. A 3-tier architecture was introduced to efficiently allocate resources in hybrid environments. This format assigns tasks to the fog layer and the remaining requests are satisfied using resources from the cloud. Certain factors including overall computational time, cost and response are taken into account.

Bee Life Algorithm is addressed for job scheduling issues and to distribute each and every task to edge and fog nodes situated at network termination. Its primary emphasis lies in shrinking both the memory usage and time taken for execution necessary for tasks performed on fog nodes [21]. Two demonstrable methodologies, one focused on cost-aware and the other on time-awareness, are employed to arrange the task scheduling across both cloud and fog resources. This implementation is mainly emphasis on fixing the scheduling concerns in hybrid conditions mainly for BoT applications.

Misra et al. [25] study focuses on task offloading issues for SDN-enabled fog networks that minimizes delay in IoT tasks & the energy usage of devices. The solution involves employing a greedy-heuristic-based approach.

3. System Model

The designed system is modelled with various interconnected fog nodes that comprises mesh topology. The fog network incorporates a collection of n number of heterogeneous fog environment FN = (F, link) where $F = \{f_1, f_2, f_3,..f_n\}$represents fog nodes and link = $\{x_{ij}|i,j \in F\}$ represents the communications connections in between fog nodes. This model consists of m independent tasks, where $T = \{t_1, t_2, t_3,....t_m\}$that are transferred from IoT devices to the fog server after a precise time duration. This system has two main components i.e task controller and resource manager. Figure 1 outlines the proposed system architecture. The task controller estimates the requirement of the task based on their predefined deadline and arrival time. If more than one task has the same deadline then the task that arrives first gets the priority, so based on this the task controller generates a task priority list and assigns that list to the resource manager for further processing. The resource manager will track the availability of resources in the network to assign the fog resources to respective tasks. Calculate the response time for the requested task utilizing equation (1) for all available fog nodes and divide them into two parts i.e DSL and USL, one that satisfies task deadline and other do not satisfy task deadline respectively.

$$R_T = D_t^{Tx} + D_t^{Proc} + D_t^{Prop} + D_t^{Que}, \quad \forall_i \in F \quad (1)$$

Here, for any task that has been submitted to the resource manager, RT represents the response time that comprises (i) transmission time D_t^{Tx}, time to transmit input file (ii) processing time D_t^{Proc} (iii) propagation delay D_t^{Prop} (iv) delay in waiting queue D_t^{Que}. The resource manager runs a scheduling algorithm for scheduling tasks to particular fog resources. Then, compute the system's energy consumption using equation (3) for every fog resource that exists in DSL, and arrange them in the descending order and create PRL. We can presently compute the proportion of IoT tasks that adhere to their deadlines. To achieve this, let DSN denote the count of tasks for which their predefined deadlines are met and DS% represents the portion of IoT tasks that adhere to their deadlines.

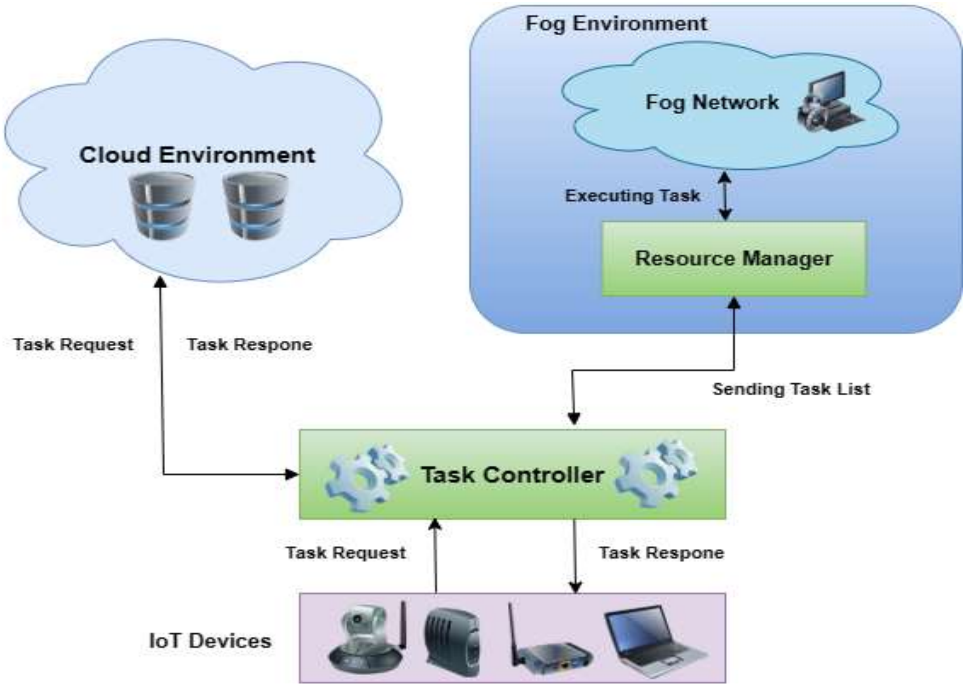

Figure 1: Proposed System Architecture - PSGS

$$DS^{\%} = DS_N/n \quad (2)$$

$$E_{con} = (T_A * F_i^{active} + F_i^{idle} * T_{idle}), \quad \forall_i \in F \quad (3)$$

Here, E_{con} represents the energy consumption by fog nodes while executing m no. of tasks. E_{con} comprises (i) T_A represents time taken for assigning task (ii) F_i^{active} represents the active time of every single fog node (iii) F_i^{idle} denotes the idle time of every single fog node (iv) T_{idle} represents the idle time of the task. To acquire the idle time for each fog node, it is essential to initially determine the makespan, which signifies the maximum execution time of a fog node within the set of all fog nodes. The equation (4) provides the value of the makespan (ɱş).

$$\text{ɱş} = \max_{\forall i \in F} \left(\sum_{\forall i \in T} (D_t^{Proc} * x_{it}) \right), \quad \forall_i \in F \qquad (4)$$

Here, (i) x_{it} equals 1, if task t_m is designated to fog node f_i (ii) x_{it} equals 0, if task t_m is not designated to fog node f_i. At last, assign the task to any fog resource randomly as shown in Algorithm 2. If no fog resource meets the deadline for a particular task, then in that scenario the task will be sent to the cloud server for ongoing processing.

4. Proposed Algorithm

To address the task scheduling constraints in the foggy environment, a novel strategy is proposed called Priority based Semi Greedy Strategy (PSGS). The proposed strategy consists of the following steps.

Algorithm 1: PSGS Task Scheduler

Input: m no. of independent task, FN = (F, *link)*, where F: set of n fog nodes, *link*: set of communication connections between fog nodes,
Output: T→F

arrange all task in ascending order of their deadline and arrival time;
obtain the availability of all fog nodes from RM;
  for each $t_i \in T$ do
  Initialize task list DSL that satisfies the deadline of t_i;
  Initialize task list USL that does not satisfies the deadline of t_i;
    for each $f_i \in F$ do
    Determine the Response Time R_t of task t_i using;
      if $R_{Ti} < t_i^{deadline}$ then
        DSL ← DSL ∪ f_i;
      else
      USL ← USL ∪ f_i;
      end if
    end for
  if DSL ≠ empty then
    call PSGS (t_i, DSL, γ)
    else
    Schedule Task t_i to the cloud;
  end if
end for
return DSL

The foremost purpose of the PSGS is to assign the requested IoT task to the respective fog resource to meet their deadline. The tasks are allocated based on their priority to diminish the energy consumption and optimizing makespan.

Algorithm 2 PSGS: Priority based Semi Greedy Strategy

Input: t_i, DSL, Size
Output: schedule Task t_i to fog resource
1: **Suppose** DSL = Size;
2: **for each** $f_i \in$ DSL do
3: **Calculate** the Energy Consumption E_{con} for all fog nodes;
4: **end for**
5: **Arrange** DSL in non-ascending preference of E_{con} and create PRL;
6: **Assign** the random fog node from PRL to Task ti;
7: **Update** f_{index} ;
8: **Update** PRL;
9: **return** scheduled t_i ;

5. Simulation Results and Analysis

Here, in this division we have enlightened the effectiveness and comparison of the introduced algorithm with the existing baseline algorithms i.e FCFS[22], EDF[23], GFE[24], Detour[25], PSG[13].

5.1. Simulation Configuration

The effectiveness of the proposed work and the simulations are carried out in C++ programming language on Dev-C++ 5.11 IDE. The experiments are coded on the device 12th Gen Intel(R) Core(TM) i5-1235U, 1.30 GHz, 16 GB of RAM. To confirm the algorithm's reliability, we run every single experiment at least 10 times and calculate its average value. In our experiment, we have performed baseline algorithms with the proposed one using a varied no. of tasks and fog nodes.

5.2. Simulation Results

To authenticate the impact of proposed algorithm with respect to the existing baseline algorithms we have considered two different scenarios (1) we have fixed the number of tasks and the impact of distinct number of fog nodes is considered and (2) we have fixed the count of fog nodes and the impact of distinct number of tasks is considered.

5.2.1 The significance of varying number of fog nodes

Figure 2, 3 and 4 presents the impact of the condition in which there is an alteration in the quantity of fog nodes and fixed quantity of tasks on the various algorithms. Here, we have fixed the number of tasks to 100 and we have taken different numbers of fog nodes like 10, 15, 20, 25 and 30. In this experiment we have observed that if the number of fog nodes increases then the overall number of tasks that meet their deadline would also increase.

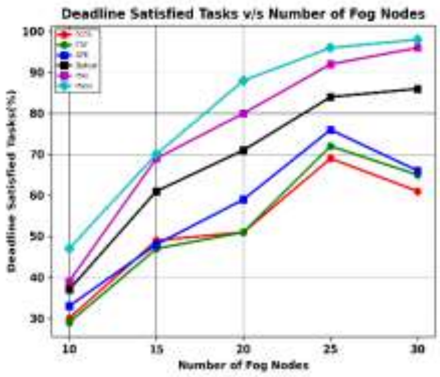

Figure 2: Deadline Satisfied Tasks vs No. of fog Nodes

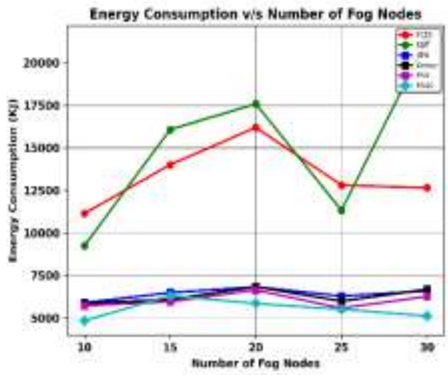

Figure 3: Energy Consumption vs No. of Fog Nodes

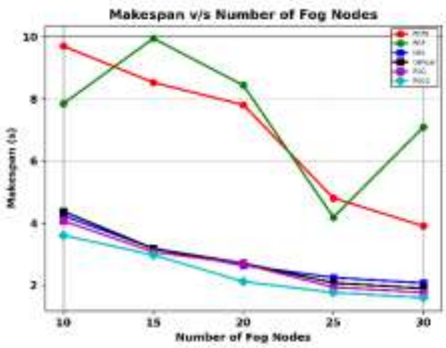

Figure 4: Makespan vs No. of Fog Nodes

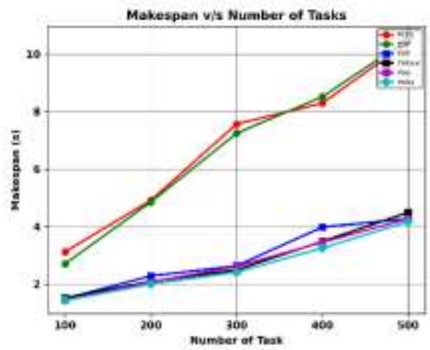

Figure 5: Makespan vs No. of Tasks

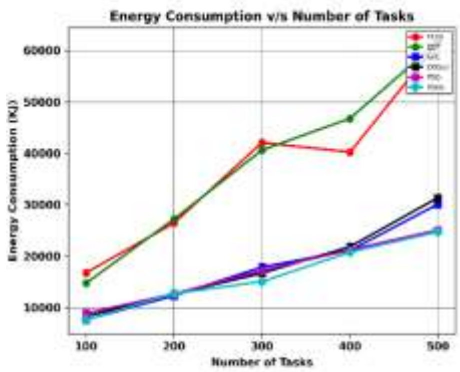

Figure 6: Energy Consumption vs No. of Tasks

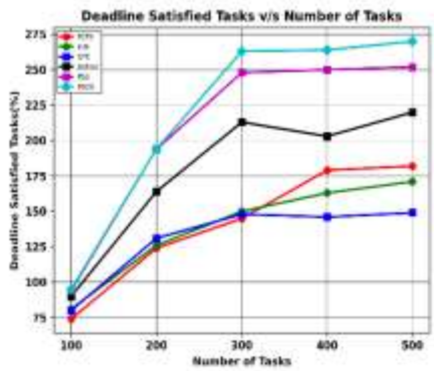

Figure 7: Deadline Satisfied Tasks vs No. of Tasks

The change in prioritizing tasks on the basis of deadline & arrival time and also, tracking available resources for efficient allocation significantly shows that the probability of tasks that fulfill their deadline surpasses the baseline algorithms. Here, if the quantity of fog nodes is 20 then our proposed PSGS fulfills 10% deadline satisfied tasks as compared to PSG and 23.9% as compared to Detour. In terms of energy consumption, the PSGS shows 18.2% higher significance and in the case of makespan of the system the PSGS shows a little difference as compared to PSG, Detour, GFE but it shows much difference when compared to FCFS and EDF.

5.2.2 The significance of varying number of tasks

Figure 5, 6 and 7 presents the impact of the condition in which there is a change in the count of tasks and fixed quantity of fog nodes on the different algorithms. Here, we have taken variation in the count of tasks i.e 100, 200, 300, 400, 500 and we have fixed number of fog nodes i.e 50. In our suggested system due to prioritizing tasks and resources there is a rise in the number of deadline satisfied tasks which is about 6% and 22% as compared to first best and Detour respectively. The rise in the number of tasks enforces an additional burden on the system, causing increased energy consumption and makespan. However, the PSGS shows better results compared to the baseline algorithms, with a 12.5 % reduction in energy consumption and a 16.1 % reduction in makespan.

6. Conclusion

Fog computing coupled with cloud is the most beneficial model for delay sensitive and real time applications. The primary objective of proposed strategy is to enhance the system's overall energy efficiency while still adhering to deadline of the task. In the given study, the main emphasis is on the task scheduling problems in fog-computing environment. The addressed task scheduling technique has considered optimization of makespan and energy usage of fog resources while fulfilling the deadline constraints of the IoT tasks. If the fog resource does not satisfy the task deadline, then that task is assigned to the cloud for processing. To achieve this objective a priority based semi greedy strategy is proposed in which tasks are scheduled on priority basis and the most coherent fog resource is assigned to that respective task. The task priority is not based only on the deadline, as done in most studies but also considering arrival time as well. The suggested approach surpasses the performance of the basic algorithms in reference to the fraction of deadline satisfied IoT tasks, optimizes the efficiency of energy usage and system's makespan. In the proposed strategy the requested tasks have been considered as the independent IoT tasks.

In the future work, the proposed strategy may consider meta-heuristic and deep learning concepts for dynamic planning of fog resources. PSGS may be enhanced to schedule IoT tasks as the dependent tasks.

References

[1] Martinez, I., Hafid, A. S., & Jarray, A.: Design, resource management, and evaluation of fog computing systems: a survey. IEEE Internet of Things Journal, 8(4), 2494-2516 (2020). DOI: 10.1109/JIOT.2020.3022699

[2] Sabireen, H., & Neelanarayanan, V. J. I. E.: A review on fog computing: Architecture, fog with IoT, algorithms and research challenges. Ict Express, 7(2), 162-176 (2021). https://doi.org/10.1016/j.icte.2021.05.004.

[3] Laroui, M., Nour, B., Moungla, H., Cherif, M. A., Afifi, H., & Guizani, M.: Edge and fog computing for IoT: A survey on current research activities & future directions. Computer Communications, 180, 210-231(2021). https://doi.org/10.1016/j.comcom.2021.09.003

[4] Li, W., Liao, K., He, Q., & Xia, Y.: Performance-aware cost-effective resource provisioning for future grid IoT-cloud system. Journal of Energy Engineering, 145(5), 04019016 (2019). https://doi.org/10.1061/(ASCE)EY.1943-7897.0000611.

[5] Kaur, N., Kumar, A., & Kumar, R.: A systematic review on task scheduling in Fog computing: Taxonomy, tools, challenges, and future directions. Concurrency and Computation: Practice and Experience, 33(21), e6432 (2021). https://doi.org/10.1002/cpe.6432

[6] Costa, B., Bachiega Jr, J., de Carvalho, L. R., & Araujo, A. P.: Orchestration in fog computing: A comprehensive survey. ACM Computing Surveys (CSUR), 55(2), 1-34 (2022). https://doi.org/10.1145/3486221

[7] Abdalla, Peshraw Ahmed, and Asaf Varol.: "Advantages to disadvantages of cloud computing for small-sized business." 2019 7th International Symposium on Digital Forensics and Security (ISDFS). IEEE, (2019). DOI: 10.1109/ISDFS.2019.8757549

[8] Alizadeh, M. R., Khajehvand, V., Rahmani, A. M., & Akbari, E.: Task scheduling approaches in fog computing: A systematic review. International Journal of Communication Systems, 33(16), e4583 (2020). https://doi.org/10.1002/dac.4583

[9] Yin, Y., Chen, L., Xu, Y., Wan, J., Zhang, H., & Mai, Z.: QoS prediction for service recommendation with deep feature learning in edge computing environment. Mobile networks and applications, 25, 391-401(2020). https://doi.org/10.1007/s11036-019-01241-7

[10] Singh, S. K., Dhurandher, S. K.: Architecture of Fog Computing, Issues and Challenges: A Review. In 2020 IEEE 17th India Council International Conference (INDICON) (pp. 1-6). IEEE (2020). DOI: 10.1109/INDICON49873.2020.9342074

[11] Gao, H., Huang, W., Yang, X., Duan, Y., & Yin, Y.: Toward service selection for workflow reconfiguration: An interface-based computing solution. Future Generation Computer Systems, 87, 298-311 (2018). https://doi.org/10.1016/j.future.2018.04.064

[12] Azizi, S., Shojafar, M., Abawajy, J., & Buyya, R.: Deadline-aware and energy-efficient IoT task scheduling in fog computing systems: A semi-greedy approach. Journal of network and computer applications, 201, 103333 (2022). https://doi.org/10.1016/j.jnca.2022.103333.

[13] Najafizadeh, A., Salajegheh, A., Rahmani, A. M., & Sahafi, A.: Multi-objective Task Scheduling in cloud-fog computing using goal programming approach. Cluster Computing, 25(1), 141-165 (2022). DOI:https://doi.org/10.1007/s10586-021-03371-8

[14] Yadav, A. M., Tripathi, K. N., & Sharma, S. C.: An enhanced multi-objective fireworks algorithm for task scheduling in fog computing environment. Cluster Computing, 1-16 (2022). DOIhttps://doi.org/10.1007/s10586-021-03481-3

[15] Kaur, N., Kumar, A., & Kumar, R.: TRAP: task-resource adaptive pairing for efficient scheduling in fog computing. Cluster Computing, 25(6), 4257-4273 (2022). DOIhttps://doi.org/10.1007/s10586-022-03641-z

[16] Wang, S., Zhao, T., & Pang, S.: Task Scheduling Algorithm based on Improved Firework Algorithm in Fog Computing. IEEE Access, 32385-32394 (2020). DOI: 10.1109/ACCESS.2020.2973758

[17] Fizza, K., Auluck, N., & Azim, A.: Improving the schedulability of real-time tasks using fog computing. IEEE Transactions on Services Computing, 15(1), 372-385 (2019). DOI: 10.1109/TSC.2019.2944360.

[18] Abdel-Basset, M., Mohamed, R., Elhoseny, M., Bashir, A. K., Jolfaei, A., & Kumar, N.: Energy-aware marine predators algorithm for task scheduling in IoT-based fog computing applications. IEEE Transactions on Industrial Informatics, 17(7), 5068-5076 (2020). DOI: 10.1109/TII.2020.3001067.

[19] Ghanavati, S., Abawajy, J., & Izadi, D.: An energy aware task scheduling model using ant-mating optimization in fog computing environment. IEEE Transactions on Services Computing, 15(4), 2007-2017 (2020). DOI: 10.1109/TSC.2020.3028575.

[20] Binh, H. T. T., Anh, T. T., Son, D. B., Duc, P. A., & Nguyen, B. M.: An evolutionary algorithm for solving task scheduling problem in cloud-fog computing environment. In Proceedings of the 9th International Symposium on Information and Communication Technology (pp. 397-404) (2018). https://doi.org/10.1145/3287921.3287984.

[21] Bitam, S., Zeadally, S., & Mellouk, A.: Fog computing job scheduling optimization based on bees swarm. Enterprise Information Systems, 12(4), 373-397(2018). https://doi.org/10.1080/17517575.2017.1304579.

[22] W. Zhao, J. A. Stankovic, Performance analysis of fcfs and improved fcfs scheduling algorithms for dynamic real-time computer systems, in: 1989 Real-Time Systems Symposium, IEEE Computer Society, 1989, pp. 156–157, (1989). DOI: 10.1109/REAL.1989.63566.

[23] J. A. Stankovic, M. Spuri, K. Ramamritham, G. C. Buttazzo, Deadline scheduling for real-time systems: EDF and related algorithms, Vol. 460, Springer Science & Business Media, (2012).

[24] J. Xu, X. Sun, R. Zhang, H. Liang, Q. Duan, Fog-cloud task scheduling of energy consumption optimisation with deadline consideration, International Journal of Internet Manufacturing and Services 7 (4) 375–392, (2020). https://doi.org/10.1504/IJIMS.2020.110228.

[25] S. Misra, N. Saha,: Detour: Dynamic task offloading in software-defined fog for Iot applications, IEEE Journal on Selected Areas in Commun11ications 37 (5) 1159–1166, (2019). DOI: 10.1109/JSAC.2019.2906793