PGA: A Priority-aware Genetic Algorithm for Task Scheduling in Heterogeneous Fog-Cloud Computing

Farooq Hoseiny*, Sadoon Azizi*, Mohammad Shojafar†, Fardin Ahmadizar‡ and Rahim Tafazolli†
* Department of Computer Engineering and IT, University of Kurdistan, Sanandaj, Iran
farooq.hoseiny@eng.uok.ac.ir {s.azizi,f.ahmadizar}@uok.ac.ir
† 6GIC/ICS, University of Surrey, Guildford, United Kingdom
{m.shojafar,r.tafazolli}@surrey.ac.uk
‡ Department of Industrial Engineering, University of Kurdistan, Sanandaj, Iran

***Abstract*—Fog-Cloud computing has become a promising platform for executing Internet of Things (IoT) tasks with different requirements. Although the fog environment provides low latency due to its proximity to IoT devices, it suffers from resource constraints. This is vice versa for the cloud environment. Therefore, efficiently utilizing the fog-cloud resources for executing tasks offloaded from IoT devices is a fundamental issue. To cope with this, in this paper, we propose a novel scheduling algorithm in fog-cloud computing named *PGA* to optimize the multi-objective function that is a weighted sum of overall computation time, energy consumption, and percentage of deadline satisfied tasks (PDST). We take the different requirements of the tasks and the heterogeneous nature of the fog and cloud nodes. We propose a hybrid approach based on prioritizing tasks and a genetic algorithm to find a preferable computing node for each task. The extensive simulations evaluate our proposed algorithm to demonstrate its superiority over the state-of-the-art strategies.**

***Index Terms*—Fog-cloud computing, Internet of things (IoT), task scheduling, multi-objective optimization, genetic algorithm.**

I. Introduction

Internet of Things (IoT) is shaping the future of connectivity, processing, and access. Any things can connect to the Internet in the IoT, including sensors, cell phones, cameras, wearables, and actuators. As a result, the number of devices connected to the Internet has increased dramatically in recent years. These devices generate large amounts of data in a short time that requires processing, storage, and networking resources [1]. Cloud computing is a centralized computing model that enables the utilization of powerful computing resources through IoT devices, and users [2]. Many IoT applications such as augmented reality, connected cars, drones, industrial robotics, and patient health monitoring systems are sensitive to delay. As a result, they must be implemented and responded to as quickly as possible [3], [4].

As respects the centralization of cloud computing leads to low and unacceptable quality of service for such applications [5]. Therefore, to reduce cloud constraints, in 2012 Cisco proposed a distributed computing model for cloud computing [6]. The fog computing paradigm has a three-tier architecture in which the fog layer contract between cloud data centers and IoT devices [7]. The fog environment includes computing, storage, and network devices called fog node [8].

Motivation. Due to the proximity of nodes to IoT devices, the delay in sending information and returning responses to applications will be much less than in cloud data centers [9]. The fog layer consists of limited resources that alone do not meet the request and needs of IoT users [8]. As a result, a new paradigm called fog-cloud computing can meet the needs of various applications. The fog-cloud computing environment has several advantages, such as reducing latency, reducing network traffic, and increasing energy efficiency. However, this new model faces many research and operational challenges. Resource allocation and task scheduling are some of the most significant challenges [2]. Scheduling issues are assigning the appropriate node to tasks according to their requirements. The scheduling problem is divided into two parts: (i) What tasks should be placed on what nodes? And (ii) What is the order of execution of tasks in each node?

Task scheduling algorithm has a significant impact on the optimal use of system resources and user satisfaction. For this reason, researchers proposed many heuristic and meta-heuristic algorithms recently [10]–[17]. However, to the best of our knowledge, none of them considered the deadline of tasks, energy consumption, and overall computation time. Thus, in this paper, we aim to jointly reduce *the total overall computation time and energy consumption of the fog-cloud computing environment.*

Our contributions. In this paper, we address the problem of computational task scheduling in heterogeneous fog-cloud computing environments. We aim to jointly minimize the total computation time and energy consumption while the percentage of tasks completed before their deadline is maximized. We summarize our main contributions as follows:

- To respect the delay-sensitivity of tasks and to efficiently utilize the fog-cloud resources, we propose a new priority-aware genetic algorithm to solve the problem.
- To achieve these goals, we introduce a deadline, length-aware classification mechanism to find a preferable environment for each task.
- Putting all this together, the proposed approach that we named *PGA* provides a very good convergence time and a significant performance.

The rest of the paper is organized as follows. The next

section studies the related literature. We describe system modeling, including system architecture and problem formulation, in Section III. The proposed approach is introduced in Section IV. In Section V, we evaluate the performance of the proposed method. Finally, we conclude our work in Section VI.

II. Related Work

Due to the significant impact of scheduling on a computing system's performance and cost, in the literature, many algorithms have been proposed that deal with this problem. In the following, we review some of the studies that have been conducted on this still-challenging issue.

The meta-heuristic algorithms are highly attended by researchers due to their searching nature in the search space. For example, the authors of [10] combine the invasive weed optimization (IWO) and the cultural evolution algorithm (CEA) to minimize the makespan and energy consumption in the fog-cloud environment. Similarly, in [11], the authors propose a hybrid method of ant colony optimization (ACO) and genetic algorithm (GA) to reduce the makespan and consumed energy of fog infrastructure. To find an optimal trade-off between energy consumption and the makespan of the system, the authors in [12] introduce a novel bio-inspired optimization strategy called ant mating optimization (AMO).

Many researchers decide to utilize the genetic algorithm (GA) due to the high efficiency and scalability for real-time applications. In [13], the authors suggest a hybrid GA algorithm to reduce the computation and response times for cloud servers. The authors have tried to provide a suitable solution for the introduced model by combining a greedy method with GA's binary coding. In [14], the authors employed the one-point crossover method to a modified non-dominated sorting genetic algorithm (NSGA-II) and introduced a multi-objective model to decrease the makespan and cost in fog-cloud environments.

Some current research works have used heuristic, fuzzy, and deep learning methods to solve the task scheduling problem in a fog-cloud environment. For example, the authors of [15] formulate the task scheduling problem as a binary nonlinear programming model to reduce the deadline violations of tasks. Then, they propose a heuristic algorithm to reduce the amount of violation time of all tasks. To improve the percentage of jobs that meet their deadline, the authors [16], [17] introduced a fuzzy priority deadline-based algorithm for scheduling tasks. In [17], the authors proposed a reinforcement learning architecture to schedule tasks on cloud servers to reduce the execution time of tasks.

Comparisons. All of the research mentioned above works have taken valuable steps to solve the scheduling problem. However, none of them jointly considered the deadline of tasks, energy consumption, and overall computation time. Moreover, most of them suffer from the high running time, which is a critical issue for delay-sensitive IoT tasks. This study considers all the objectives above and proposes an approach that provides a feasible and practical solution for the task scheduling problem with a reasonable running time.

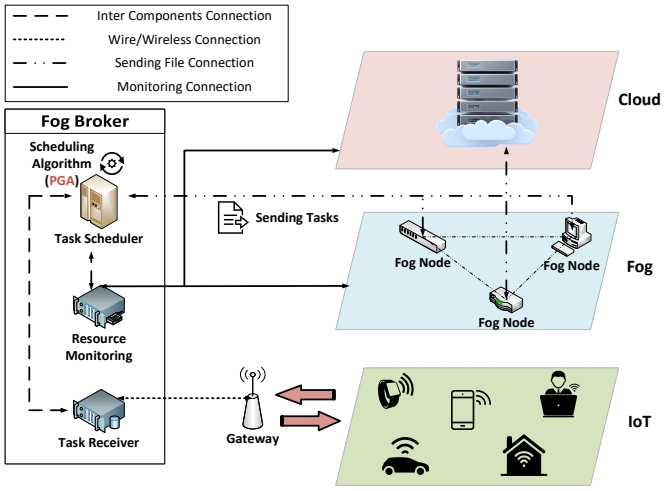


Fig. 1: An illustration of the IoT-Fog-Cloud architecture.

III. System Model

This section introduces IoT-Fog-Cloud architecture (§III-A) and formulates the task scheduling problem (§III-B).

A. System Architecture

Processing IoT requests at the fog nodes have restriction relating to the low computational resource. Besides, it incurs high delays for users and impressive energy consumption for the servers to using cloud computing. To efficiently utilization of existing infrastructure, we consider a fog-cloud environment for processing requests from IoT users, as shown in Fig. 1.

According to the devices and applications available in the first layer, users send diverse requests through gateways to the fog broker on the IoT layer's edge. Fog broker consists of three separate and connected components: task receiver, task scheduler, and resource monitoring. The duties of the first component include several operations. In the first step, requests are received and decomposed into tasks [18]. Then, the task receiver estimates data about tasks contains the number of instructions and deadline. Nodes are monitored and evaluated by the resource monitoring component. In other words, this component periodically provides a report on the status of resources to the task scheduler unit.

The task scheduler is the principal component of the fog broker. We require data and information to stack up in the task scheduler to run the algorithm. The algorithm correspondent to users' requirements and the system's status tries to find the optimal solution.

B. Problem Formulation

Here, we first describe our objectives, including overall computation time, energy consumption, and Percentage of the Deadline Satisfied Tasks (PDST). After that, we present our multi-objective optimization model.

Let $T = \{T_1, T_2, T_3, \ldots, T_n\}$ be the set of all tasks at the task scheduler. For each task T_i, we consider two key

attributes, called length (T_i^{len}) and deadline (T_i^{dead}). Also consider a fog-cloud system with $N = \{N_1, N_2, N_3, \ldots, N_m\}$ heterogeneous computation nodes, where each node N_j has four main attributes, called CPU processing rate (N_j^{cpu}), communication delay from the Fog Broker (N_j^{del}), power consumption for active mode (N_j^{act}) and idle mode (N_j^{idl}). We also denote $X_{(n\times m)}$ as a decision matrix. $x_{ij} = 1$ means that task T_i is assigned to node N_j; otherwise, $x_{ij} = 0$.

Computation Time. To model the overall computation time of the T tasks, we should first obtain the execution time of each task T_i on node N_j, which is calculated as follows:

$$\mathcal{E}_{ij} = \frac{T_i^{len}}{N_j^{cpu}} \qquad T_i \in T,\, N_j \in N, \tag{1}$$

The overall computation time of the system to process all n tasks, can be determined as follows:

$$\mathfrak{C}^{comp} = \sum_{j=1}^{m}\sum_{i=1}^{n} \mathcal{E}_{ij} \times x_{ij}\,, \tag{2}$$

Energy Consumption. During the execution of all the T tasks, the energy consumption of each node N_j is dependent on two main factors: its state mode, i.e., active or idle, and how long it is in that state. The active time of a node N_j is equal to the summation of the processing time for all assigned tasks. In mathematical words:

$$A_j = \sum_{i=1}^{n} \mathcal{E}_{ij} \times x_{ij}\,, \tag{3}$$

Now, let M be the makespan of the system, the time when the first task starts its execution to the time when the last task is executed:

$$M = \max_{1\le j\le m} (A_j)\,, \tag{4}$$

Based on eq. (3) and (4), the energy consumption of node N_j can be calculated as:

$$\mathfrak{E}_j = A_j \times P_j^{act} + (M - A_j) \times P_j^{idle}\; j \in \{1, \ldots, m\}, \tag{5}$$

Therefore, the total energy consumption can be expressed as:

$$\mathfrak{E}^{\mathsf{TOT}} = \sum_{j=1}^{m} \mathfrak{E}_j\,. \tag{6}$$

PDST. To obtain the PDST of the system, we first should calculate the response time of each task T_i, which is consists of its execution time, communication time from the fog broker to the assigned node, and waiting time, i.e.,

$$R_i = \sum_{j=1}^{m} \left(2 \times N_j^{del} + \mathcal{E}_{ij} + W_{ij}\right) \times x_{ij}\,, \tag{7}$$

Let P denote the set of deadline satisfied tasks, i.e., those tasks for which $R_i \le T_i^{dead}$. Hence, PDST is equal to

$$PDST = \frac{|P|}{n - |P|}\,, \tag{8}$$

C. Overall Objective Function

Our overall objective function is mapping tasks to heterogeneous nodes so that the overall computation time, energy consumption, and PDST are minimized.

$$\eta = w_1 \times \mathfrak{C}^{comp} + w_2 \times \mathfrak{E}^{\mathsf{TOT}} + w_3 \times (1 - PDST)\,. \tag{9}$$

where w_k, $k \in \{1, 2, 3\}$ are weight coefficients to accentuate the role of the *three* considered objectives.

Therefore, the optimization problem is expressed as follows.

$$\min \quad \eta \tag{10a}$$

$$\text{s.t.} \quad x_{ij} \in \{0, 1\},\, j \in \{1, \ldots, m\}, i \in \{1, \ldots, n\}, \tag{10b}$$

$$\sum_{j=1}^{m} x_{ij} = 1, \quad i \in \{1, \ldots n\}, \tag{10c}$$

$$\sum_{k=1}^{3} w_k = 1, \tag{10d}$$

To normalize the model, the lowest value of computation time and energy consumption must be calculated. The lower bound for computation time is obtained when each task is allocated to the most powerful node. Hence,

$$\mathfrak{C}_{min}^{comp} = \frac{\sum_{i=1}^{n} T_i^{len}}{\max_{1\le j\le m}(N_j^{cpu})}\,. \tag{11}$$

To obtain a lower bound for energy consumption, we first calculate the minimum required time to execute all tasks, i.e., minimum makespan, in fog-cloud infrastructure, which will be defined as follows [2]:

$$M_{min} = \frac{\sum_{i=1}^{n} T_i^{len}}{\sum_{j=1}^{m} N_j^{cpu}}\,, \tag{12}$$

Then, we assume the least active power consumption for all nodes. Therefore, minimum energy consumption can be determined as follows:

$$\mathfrak{E}^{\mathsf{TOT}} = m \times \min_{1\le j\le m} (P_j^{max}) \times M_{min}\,, \tag{13}$$

Consequently, the normalized objective function can be expressed as below:

$$\tilde{\eta} = \tilde{w_1} \times \frac{\mathfrak{E}_{min}^{\mathsf{TOT}}}{\mathfrak{E}^{\mathsf{TOT}}} + \tilde{w_2} \times \frac{\mathfrak{C}_{min}^{comp}}{\mathfrak{C}^{comp}} + \tilde{w_3} \times PDST\,. \tag{14}$$

Now our optimization problem (10a) becomes

$$\max \quad \tilde{\eta} \tag{15a}$$

$$\text{s.t.} \quad x_{ij} \in \{0, 1\},\, j \in \{1, \ldots, m\}, i \in \{1, \ldots, n\}, \tag{15b}$$

$$\sum_{j=1}^{m} x_{ij} = 1, \quad i \in \{1, \ldots n\}, \tag{15c}$$

$$\sum_{k=1}^{3} \tilde{w}_k = 1. \tag{15d}$$

IV. Proposed Algorithm

In this section, we propose our hybrid scheduling algorithm, PGA, for achieving the qualified situation of the system and the satisfaction of users. Algorithm 1 contains two interrelated parts: classification (§IV-A) and genetic algorithm (§IV-B).

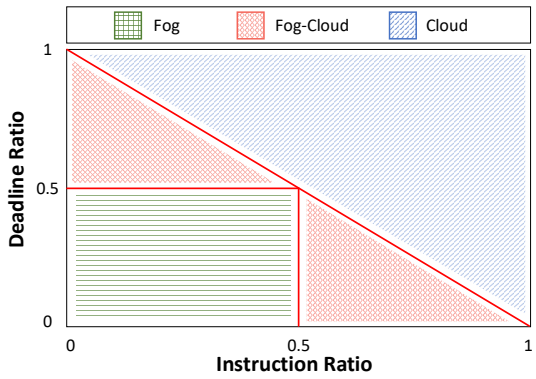


Fig. 2: Classification of heterogeneous tasks over Cloud, Fog, and Fog-Cloud based on deadline and instruction ratios.

A. Task Classification

This section intends to classification tasks according to their specific characteristics and prioritizes tasks with low deadlines. Therefore, we first sort tasks in ascending order based on their deadline to prioritize the execution of the task on the node. Next, we investigate the tasks situation according to their attributes universally. For this purpose, we define two parameters as follows:

$$\mathbf{R}_i^{\text{dead}} = \frac{T_i^{dead}}{\max_{1\leq x\leq n}\left(T_x^{dead}\right)}, \tag{16}$$

$$\mathbf{R}_i^{\text{len}} = \frac{T_i^{len}}{\max_{1\leq x\leq n}\left(T_x^{len}\right)}, \tag{17}$$

$\mathbf{R}_i^{\text{dead}}$, $\mathbf{R}_i^{\text{len}}$ are the ratio coefficient to exhibit the location of T_i on the chart ranging (0,1). In this way, the tasks are thought-out against each other then we can effortlessly classify them. Fig. 2 shows which layer is convenient to satisfy the task requirement. The points with a low ratio are closer to the origin of the coordinates and belong to the class 1. The class 2 contains tasks that have more deadlines to execute or require extra powerful computing resources. According to our study, we observed that the tasks in the red region significantly affect the energy consumption and computation time of the system. Therefore, representatives of fog and clouds node are selected randomly for an immediate decision. According to the calculated energy consumption, the best class is determined to process the desired task. We obtained the determination of classification by performing many experiments.

Algorithm 1 presents the proposed algorithm (PGA) for the task scheduling problem in fog-cloud infrastructure. As mentioned before, the algorithm sorts the task in ascending order based on the deadline (line 1). Then, constructs two sets to classify tasks (lines 2,3). In the next step, we compute the deadline and instruction ratio for each task to obtain the fog and cloud list (lines 4 to 22). Finally, the genetic algorithm runs in parallel for fog and cloud list (lines 23,24). The proposed PGA has some modifications from the original GA. First, instead of generation a random initial population, we first sort tasks according to their deadline. Second, for crossover

Algorithm 1 PGA algorithm

Input: T, N
Output: Allocated T on N
1: $AscSort(T, T_i^{dead})\ \forall i \in N$
2: $fogList \leftarrow \{\}$
3: $cloudList \leftarrow \{\}$
4: **for all** T_i in T **do**
5: Compute $\mathbf{R}_\mathbf{i}^{\text{dead}}$ using (16)
6: Compute $\mathbf{R}_\mathbf{i}^{\text{len}}$ using (17)
7: $y = 1 - \mathbf{R}_\mathbf{i}^{\text{len}}$
8: **if** $(\mathbf{R}_\mathbf{i}^{\text{dead}} \leq 0.5) \&\& (\mathbf{R}_\mathbf{i}^{\text{dead}} \leq 0.5)$ **then**
9: $fogList \leftarrow fogList \cup \{T_i\}$
10: **else if** $(\mathbf{R}_\mathbf{i}^{\text{dead}} > y)$ **then**
11: $cloudList \leftarrow cloudList \cup \{T_i\}$
12: **else**
13: $rF \leftarrow SelectRandom(Fog)$
14: $rC \leftarrow SelectRandom(Cloud)$
15: Compute the energy consumption of rF and rC using (5)
16: **if** $(\mathfrak{E}_{rF} \leq \mathfrak{E}_{rC})$ **then**
17: $fogList \leftarrow fogList \cup \{T_i\}$
18: **else**
19: $cloudList \leftarrow cloudList \cup \{T_i\}$
20: **end if**
21: **end if**
22: **end for**
23: **call** $geneticAlgorithm(fogList)$
24: **call** $geneticAlgorithm(cloudList)$
25: **return** Allocated T on N

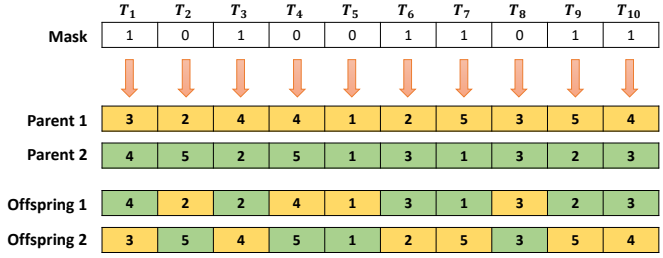


Fig. 3: Mask application procedure on PGA.

operation, we use a binary mask procedure, which is described in the following subsection.

B. PGA: Proposed Priority-based Genetic Algorithm

Crossover and mutation are the essential sections of the genetic algorithm (GA). Selecting the appropriate procedure for the crossover operator affects GA performance. In this paper, we use a binary mask vector to produce offsprings [19]. In each iteration, the crossover operator selects the first parent by the roulette wheel selection procedure and then generates the second parent and a new mask vector randomly. The binary mask operator assigns the parent gens to offspring as Algorithm 2. In the mask vector, bit 1 means swapping the genes between parents and bit 0 means copy the gens without swapping. Fig. 3 presents mask application phase with 10 tasks and 5 nodes.

V. Performance Evaluation

In this section, we conduct three experiments to evaluate our policy in various aspects and compare it against Power of 2 Choices (Po2C) [20] and Ant-Mating Optimization (AMO) algorithms [13]. The reason behind selecting these methods is that they are recent holistic fast converge algorithm that

Algorithm 2 Mask operator algorithm

Input: $Parent1$, $Parent2$, $Mask$
Output: $offspring1$, $offspring2$

copy $Parent1$ to $offspring1$
copy $Parent2$ to $offspring2$
for all i in $\{1, \ldots, n\}$ **do**
  if $(mask(i) == 1)$ **then**
    swap $offspring1(i)$ to $offspring2(i)$
  end if
end for
return $offspring1$, $offspring2$

TABLE I: Simulation settings; CN:= cloud node; FN:= fog node: Het.:= heterogeneous.

Experiment	Purpose	Parameters		
		Tasks	FN	CN
1	Convergence	250	45	15
2	Het. Tasks	[100:100:500]	45	15
3	Het. Nodes	250	[15:15:75]	[5:5:25]

chooses a node with the shortest response time for the desired task.

A. Simulation Setup

We program the PGA algorithm in Matlab 2018b. The experiments are performed on a PC with Windows 10, Intel(R) Xeon(R) CPU E7-4850 v4 with two 2.10 GHz processors and 12 GB RAM. The experiment parameters are summarized in Table I. We assess our paper into three scenarios. In the first scenario, we investigate the convergence of the proposed algorithm in fog and cloud environments separately. We conduct the first set of experiments (in simple, experiment 1) with 1000 iterations and the various number of initial populations: {50,100,200,400}. Our obtained results average based on ten runs. The second and third scenarios' purpose is to investigate the impact of the varying number of tasks and nodes, respectively. In the third experiment for each dataset, 25% of all nodes belong to the cloud environment, and 75% of the remaining nodes belong to the fog layer. We performed experiments 2 and 3 with five different datasets, and the plots are presented as averages. Tables II and III show the characteristics of tasks and nodes in detail. In Table II, the tasks are divided into three categories. Also, the size of each task is commensurate with its deadline for each category.

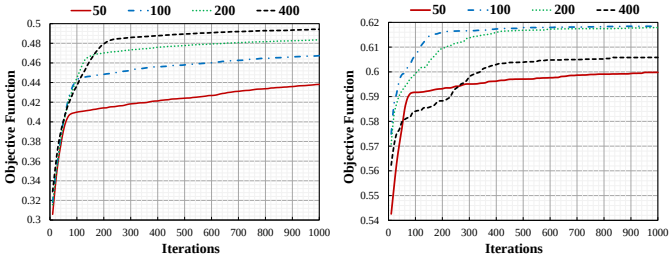
TABLE II: Task characteristics.

Parameter	Unit	Values		
		Type1	Type2	Type3
Length	MI	[100,372]	[1028, 4280]	[5123, 9784]
Deadline	s	[0.1,0.5]	[0.5, 2.5]	[2.5,10]

TABLE III: Node characteristics.

Parameter	Unit	Values	
		Fog	Cloud
CPU Rate	$MIPS$	[500, 2000]	[3000, 10000]
Max Power	W	[80, 130]	[150, 500]
Min Power	W	[60, 70]% Max Power	
Delay	s	[0.001, 0.01]	[0.1, 0.5]

Simulation metrics. In this paper, we propose a multi-objective model. In addition to the objectives, we evaluate each of the second and third experiments with makespan. Our studies show that makespan is closely related to energy. Thus, to efficiently assess the proposed algorithm, four plots are reported in each experiment: goal function (objective of η), energy consumption, computation time, and PDST.



(a) Fog (b) Cloud

Fig. 4: Convergence experiment results.

B. Results

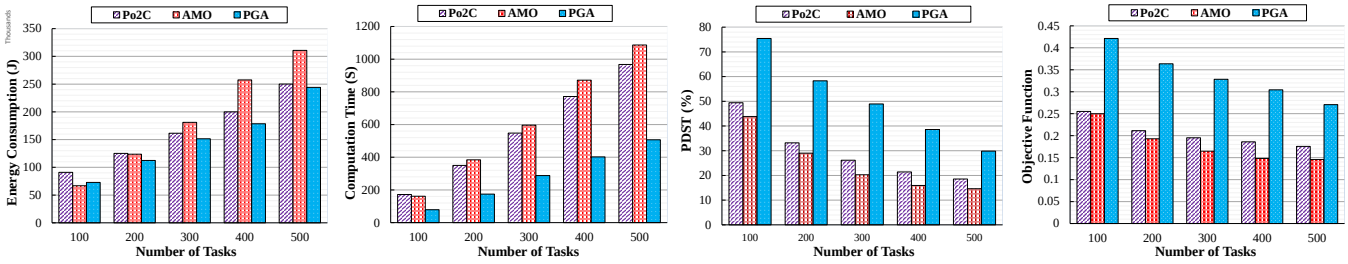
Experiment one (Heuristic convergence). Fig. 4 presents the result of the convergence time in the fog and cloud environment. The classification phase and the sorting of tasks confirm that the search space reduces and results in a good convergence time in both cloud and fog environments. Focusing on Fig. 4a, the best value for objective function is achieved when the population size and the number of iterations are set to 400 and 200, respectively. However, the best result is obtained for the cloud environment using the population size of 100 and 200 iterations (Fig. 4b). Therefore, we used these settings for the proposed algorithm in the following experiments.

Experiment two (Task trade-off). In the second set of experiments, we discuss the impact of various tasks on the algorithm performance. We present the result in Fig. 5. The results confirm that the system efficiency reduces with increment tasks. Specifically, Figs. 5a and 5d illustrate that energy consumption and makespan did not diverge significantly for all three methods, and our proposed method performed better. Same-wise, Figs. 5b and 5c indicate that the PGA results improved the computation and PDST by 52.91% and 111.6% compared to AMO.

Experiment three (Node trade-off). Fig. 6 shows the results for the impact of various nodes. As can be seen from the results, as the number of nodes increases, the algorithm performance improves. Enhancing the number of nodes increases energy consumption. Our proposed algorithm reduces the process of increasing energy consumption by balancing the load in the system. As it is explicit from the performed experiments, the proposed policy can establish an optimal solution over other algorithms and can conduct effective compared to algorithms. Fig. 6d illustrates the PGA's goal function values compared to Po2C and AMO.

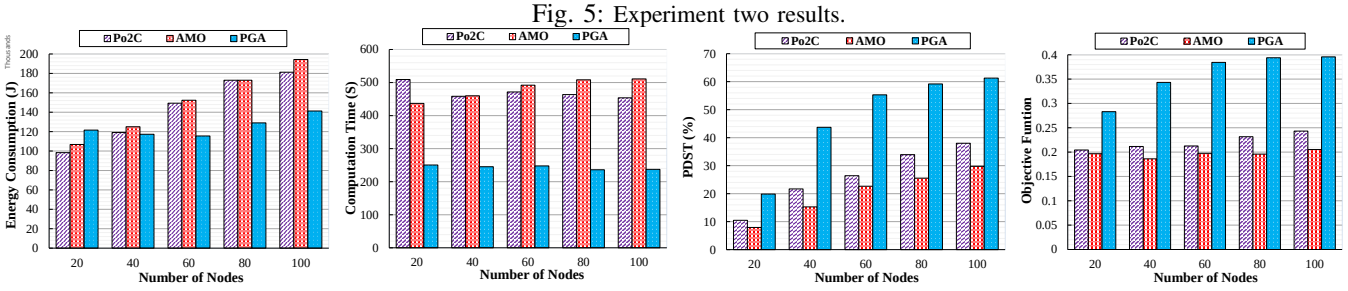
VI. Conclusions and Future Direction

This paper studied the task scheduling in fog-cloud computing systems that consist of heterogeneous computing nodes



(a) Energy Consumption (b) Computation Time (c) PDST (d) Objective η

Fig. 5: Experiment two results.



(a) Energy Consumption (b) Computation Time (c) PDST (d) Objective η

Fig. 6: Experiment three results.

with different computational capabilities. We considered the attributes of tasks, including deadline and the number of instructions, to classify them and find a desirable environment for each task. Then we proposed a priority-aware genetic algorithm, PGA, to jointly optimize total computation time, energy consumption, and the percentage of tasks completed before their deadline. We verified the efficiency of the proposed algorithm against state-of-the-art methods. The results showed that the proposed approach provides a good convergence time and significantly performs better than the compared algorithms. Our next research plan is to consider the dependency among tasks and the network topology between the fog nodes and extend our method to the various serverless IoT applications.

REFERENCES

[1] A. Yousefpour *et al.*, "Fogplan: a lightweight qos-aware dynamic fog service provisioning framework," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5080–5096, 2019.

[2] B. M. Nguyen *et al.*, "Evolutionary algorithms to optimize task scheduling problem for the iot based bag-of-tasks application in cloud–fog computing environment," *Applied Sciences*, vol. 9, no. 9, p. 1730, 2019.

[3] C. C. Byers, "Architectural imperatives for fog computing: Use cases, requirements, and architectural techniques for fog-enabled iot networks," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 14–20, 2017.

[4] R. S. Montero, E. Rojas, A. A. Carrillo, and I. M. Llorente, "Extending the cloud to the network edge." *IEEE Computer*, vol. 50, no. 4, pp. 91–95, 2017.

[5] S. Javanmardi, M. Shojafar, V. Persico, and A. Pescape, "Fpfts: A joint fuzzy pso mobility-aware approach to fog task scheduling algorithm for iot devices," *Software Practice and Experience*, 2020.

[6] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, Jan. 2012, pp. 13–16.

[7] H. O. Hassan, S. Azizi, and M. Shojafar, "Priority, network and energy-aware placement of iot-based application services in fog-cloud environments," *IET Communications*, vol. 14, no. 13, pp. 2117–2129, 2020.

[8] R. K. Naha *et al.*, "Deadline-based dynamic resource allocation and provisioning algorithms in fog-cloud environment," *Future Generation Computer Systems*, vol. 104, pp. 131–141, 2020.

[9] X.-Q. Pham *et al.*, "A cost-and performance-effective approach for task scheduling based on collaboration between cloud and fog computing," *International Journal of Distributed Sensor Networks*, vol. 13, no. 11, 2017.

[10] P. Hosseinioun, M. Kheirabadi, S. R. K. Tabbakh, and R. Ghaemi, "A new energy-aware tasks scheduling approach in fog computing using hybrid meta-heuristic algorithm," *Journal of Parallel and Distributed Computing*, 2020.

[11] X. Ren, Z. Zhang, and S. M. Arefzadeh, "An energy-aware approach for resource managing in the fog-based internet of things using a hybrid algorithm," *International Journal of Communication Systems*, vol. 34, no. 1, p. e4652, 2020.

[12] S. Ghanavati, J. H. Abawajy, and D. Izadi, "An energy aware task scheduling model using ant-mating optimization in fog computing environment," *IEEE Transactions on Services Computing*, 2020.

[13] Z. Zhou, F. Li, H. Zhu, H. Xie, J. H. Abawajy, and M. U. Chowdhury, "An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments," *Neural Computing and Applications*, vol. 32, no. 6, pp. 1531–1541, 2020.

[14] I. M. Ali *et al.*, "An automated task scheduling model using non-dominated sorting genetic algorithm ii for fog-cloud systems," *IEEE Transactions on Cloud Computing*, 2020.

[15] B. Wang, Y. Song, C. Wang, W. Huang, and X. Qin, "A study on heuristic task scheduling optimizing task deadline violations in heterogeneous computational environments," *IEEE Access*, vol. 8, pp. 205 635–205 645, 2020.

[16] C. D. S. Rajan, "Design and implementation of fuzzy priority deadline job scheduling algorithm in heterogeneous grid computing," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–8, 2020.

[17] T. Dong, F. Xue, C. Xiao, and J. Li, "Task scheduling based on deep reinforcement learning in a cloud manufacturing environment," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 11, p. e5654, 2020.

[18] J. Xu, Z. Hao, R. Zhang, and X. Sun, "A method based on the combination of laxity and ant colony system for cloud-fog task scheduling," *IEEE Access*, vol. 7, pp. 116 218–116 226, 2019.

[19] A. J. Umbarkar and P. D. Sheth, "Crossover operators in genetic algorithms: a review." *ICTACT journal on soft computing*, vol. 6, no. 1, 2015.

[20] F. Hoseiny, S. Azizi, and S. Dabiri, "Using the power of two choices for real-time task scheduling in fog-cloud computing," in *2020 4th International Conference on Smart City, Internet of Things and Applications (SCIOT)*. IEEE, 2020, pp. 18–23.