IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 16 September 2022, accepted 12 October 2022, date of publication 19 October 2022, date of current version 26 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3215555

RESEARCH ARTICLE

SMRETO: Stable Matching for Reliable and Efficient Task Offloading in Fog-Enabled IoT Networks

USMAN MAHMOOD MALIK[1,2], MUHAMMAD AWAIS JAVED[2], (Senior Member, IEEE), JAROSLAV FRNDA[3], (Senior Member, IEEE), AND JAN NEDOMA[4], (Senior Member, IEEE)

[1]Department of Computer Software Engineering, National University of Science and Technology (NUST), Islamabad 45550, Pakistan
[2]Department of Electrical and Computer Engineering, COMSATS University Islamabad (CUI), Islamabad 45550, Pakistan
[3]Department of Quantitative Methods and Economic Informatics, Faculty of Operation and Economics of Transport and Communication, University of Zilina, 01026 Zilina, Slovakia
[4]Department of Telecommunications, Faculty of Electrical Engineering and Computer Science, VSB-Technical University of Ostrava, 70800 Ostrava, Czech Republic

Corresponding author: Muhammad Awais Javed (awais.javed@comsats.edu.pk)

This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic conducted by VSB-Technical University of Ostrava, Czechia, under Grant SP2022/18.

ABSTRACT Fog computing is a key technology that supports timely and efficient computation of different tasks in IoT networks. By using the nearby fog nodes for quick task computation, application related decisions by IoT devices can be taken within the delay requirements. Resource allocation in terms of task placement on the free computing resources of the fog nodes is a major challenge in IoT networks. In this paper, we consider task offloading from IoT devices to the logically partitioned fog computing resources known as Virtual Resource Units (VRUs) to reduce the number of task outages and energy consumption of the IoT and fog nodes. We propose a two phased task offloading algorithm to minimize the number of task outages. In the first phase, we utilize the task deadline to compute the minimum number of resources required for a task from the fog nodes. To meet the heterogeneous task computing requirements, we introduce the concept of variable sized VRUs in the fog nodes. Moreover, we propose a modified Deferred Acceptance Algorithm (DAA) for stable matching between IoT tasks and variable sized VRUs. In the second phase of the algorithm, the unmatched fog node resources are distributed among the previously matched IoT tasks. Simulation results show that the proposed algorithm outperforms available techniques in the literature in terms of task outages and energy efficiency.

INDEX TERMS Fog computing, IoT, task offloading.

I. INTRODUCTION

The development of fully autonomous systems using Artificial Intelligence (AI) and learning techniques has been the primary focus of research over the last decade [1], [2]. This has resulted in quantum advances in wireless communications, advanced sensing and the Internet of Things (IoT). These advancements have revolutionized the way humans and machines interact with each other. From light bulbs acting as fully automatic computers to 3D printing of heart tissues [3], [4], we are experiencing examples of human-machine interaction personalised to our individual needs [5], [6], [7], [8].

Future IoT applications will be extremely demanding in terms of data rate, reliability, and connectivity. These demands paved the way for the emergence of Fifth Generation (5G) communication systems, and while 5G is still in the implementation phase, researchers have begun to envision the next generation of communication networks, namely 6G [9]. With these technologies, billions of devices will be connected to each other over the Internet [10]. This will evolve the communication focus from ubiquitous connectivity to automated

The associate editor coordinating the review of this manuscript and approving it for publication was Alessio Vecchio.

and intelligent connectivity [11], necessitating de-centralized computation closer to the network edge [12].

Fog computing is a decentralised computing architecture [13], [14], [15] commonly associated with IoT technology. The basis of fog computing is to overcome the drawbacks of cloud computing such as centralised computing architecture and usually large distance between cloud servers and IoT devices. With cloud computing, it is not possible to maintain the required Quality of Service (QoS) for all the applications. On the contrary, fog computing acts as an intermediate between the cloud and the IoT devices [16] bringing computing, storage, and network resources closer to the IoT devices. This enables IoT devices to compute tasks at fog nodes with low latency while also reducing their transmission energy consumption.

Although fog computing has many benefits for IoT applications, there are a few QoS challenges that require special attention. Latency improvement and energy efficiency continue to be the primary features of QoS in fog computing; however, demand for additional QoS features such as reduced task outages to improve network utilisation and energy efficiency, has gained attention from many researchers in recent years [17].

When a task suffers from outage, QoS suffers, and users lose faith in the system's ability to provide continuous services. Task outages can cause serious problems, particularly in the fields of health care and industrial IoT, where the information generated by each sensor device is critical and failure to complete the required task within the delay threshold can have serious repercussions. Task outages can occur for a variety of reasons, including scarcity of appropriate resources at the fog node, an unexpected delay in task transmission, and a failure to allocate resources while considering a task's end-to-end resource requirement.

Matching theory is an effective mathematical tool for modelling and solving a wide variety of fog computing task offloading problems. Matching theory is a simple algorithm that logically divides a fog node's available computation resources into fixed-sized Virtual Resource Units (VRUs) and matches these VRUs to IoT device tasks, ensuring that all players' objectives are met [18]. Fixed-sized VRUs simplify the matching process but have the drawback of providing all matched tasks with the same resources from a fog node. The number of VRUs with a fog node determines the maximum number of tasks that can be served by a fog node.

We propose in this paper that by using variable-sized VRUs rather than fixed sized VRUs tailored to specific resource requirements of IoT device tasks, we can greatly increase the number of useful VRUs per fog node. As a result, a fog node will be able to serve more tasks with the same resources, reducing task outages. Variable-sized VRUs pose a serious matching problem because the total number of VRUs formed from fog node computation resources cannot be predicted in advance due to their variable size. This generates a unique matching challenge in which the total capacity of a fog node to accept the number of tasks is unknown until the matching process is completed. To address this issue, we propose SMRETO, a novel many-to-one matching algorithm that is a modified implementation of the Deferred Acceptance Algorithm (DAA) [19].

TABLE 1. Literature review and contribution of this paper.

Ref	Latency	Energy Efficiency	Task Outage	Matching Theory	Variable VRU
[20]	×	✓	✓	×	×
[21]	✓	×	✓	×	×
[22]	×	✓	✓	×	×
[23]	×	✓	✓	×	×
[24]	✓	×	✓	×	×
[17]	×	✓	✓	✓	×
[25]	×	✓	✓	✓	×
[18]	✓	×	×	✓	×
[26]	×	✓	×	✓	×
[27]	✓	×	✓	✓	×
SMRETO	×	✓	✓	✓	✓

II. RELATED WORK

In this section, we discuss our literature review in the fog computing environment, with a focus on reducing task outages through latency or energy improvement with and without using matching techniques for resource allocation. Our emphasis will be on comprehending how fog node computation resources are logically partitioned into Virtual Resource Units (VRUs) and then allocated to IoT device tasks using matching theory. The parameters used to compare different offloading techniques are summarised in Table 1.

The number of accepted tasks decreases because the end-to-end resource requirement of a sensor device task is not considered when making resource allocation decisions. As a result, Jiang et al. [20] propose an energy-efficient offloading decision mechanism that ensures no task outage of any accepted task by considering both: (1) the task's end-to-end processing time and, (2) tasks already held in the fog node queue. Non-availability of sufficient resources with the fog nodes and their poor cite planning also contribute to a decrease in the number of accepted tasks, as this places an additional load on network resources in order to complete targeted tasks in time.

Wu et al. [21] achieve latency improvement through forward deployment of computing servers to encourage fog nodes to offload their maximum tasks to these servers. Forward deployment of resources also significantly reduces the resources required to achieve latency, increasing the number of accepted tasks. Omoniwa et al. [22] achieved energy efficiency in Wireless Sensor Network (WSN) by using fixed and mobile fog nodes to relay sensor node data to main fog node/cloud servers. The mobile relay fog nodes adjust their location (i.e., it addresses the fog node cite planning problem) to increase transmission energy and ensure no task outages.

Silva et al. [23] solve the fog node location and resource planning problem to improve energy efficiency and increase the number of accepted tasks. It accomplishes this by ensuring the availability of required resources at the fog node and

by processing the maximum number of tasks and applications in the fog node.

Kyung in [24] formulated a prioritised task distribution scheme that categorises incoming task requests as delay sensitive or delay insensitive. Delay sensitive tasks are processed by static fog nodes to meet the delay requirements, while delay insensitive tasks are processed by opportunistic fog nodes to relieve load on the fog nodes. This reduces task latency and outages.

On other hand, Swain et al. [17] and Chittaranjan et al. [25] formulated the offloading problem as one-to-many matching game in order to achieve energy efficiency and reduce outages. They both reduced task outages by taking task deadlines into account while formulating fog node preference profiles. As a result, fog nodes prefer tasks with short deadlines. increasing the number of accepted tasks at the fog nodes. Chiti et al. [18] used a one-to-many matching game with externalities and DAA to improve latency by accounting for queuing delay at fog nodes. They used fixed-sized VRUs with fog nodes to match fog node resources to tasks. Wu et al. [26], on the other hand, used a one-to-many matching game with the market matching concept to achieve energy efficiency in the network. In this concept, fog nodes act as vendors and tasks act as the buyers, with prices fluctuating until a stable market situation is reached. Fixed sized VRUs are used and IoT devices make their own matching decisions.

Swain et al. [27] formulated the offloading problem as a one-to-many matching game with minimum and maximum quotas to achieve balanced task assignments to fog nodes while minimising task completion time and outages. He used multi-stage DAA, in which initial matching is done with the minimum quota of each fog node to ensure load balancing. If any tasks remain, DAA is run again with the maximum fog node quota to match these tasks only.

The research on task outages in non-matching and matching based resource allocation appears to take two distinct approaches: non-matching based techniques attempt to reduce task outages by reducing resource requirements from fog nodes, allowing them to serve a greater number of tasks with the same resources, whereas matching based techniques attempt to reduce task outages by formulating preference profiles in such a way that the number of accepted tasks at fog nodes is maximised. In our review of the literature, we found no work that used the matching technique to do resource allocation in fog computing for minimum computation resource requirements of tasks in order to minimise task outages. The reason for this gap is obvious: the matching technique lacks the ability to allocate resources for specific resource requirements of tasks in fog computing.

In this paper, we propose a novel concept of variable-sized VRUs with fog nodes to make the resource allocation process flexible in the many-to-one matching algorithm. These VRUs are sized based on the task's exact resource requirements. Only with variable-sized VRUs, a many-to-one matching algorithm can allocate computation resources to tasks based on their exact requirements. This conserves valuable fog node computation resources, allowing a fog node to serve more tasks with the same resources. With the introduction of variable-size VRUs, the scope of many-to-one matching applications to resource allocation problems will be expanded.

A. CONTRIBUTIONS OF THE PAPER

In a nutshell, the contributions of this article can be summarized as follows:

1) To reduce task outages and achieve energy efficiency, we formulated the resource allocation problem in fog computing as a many-to-one matching problem.
2) We introduced a novel concept of variable-sized VRUs with fog nodes to increase the number of useful VRUs per fog node and reduce task outages. Variable sized VRUs are tailored to the specific resource requirements of tasks, allowing for more efficient use of fog node computation resources.
3) The use of variable sized VRUs generates a unique matching challenge in which the total capacity of a fog node to accept the number of matches is unknown until the matching process is completed. To address this issue, we propose SMRETO, a novel many-to-one matching technique that generates stable matching assignments.
4) Variable-sized VRUs will make the resource allocation process more flexible, broadening the scope of matching theory applications to resource allocation problems in fog computing.

The remainder of this paper is organized as follows. Section III gives system model in detail and formulates the problem statement. In Section IV, we present the proposed SMRETO algorithm, which is theoretically analyzed for stability in Section V. In Section VI, we evaluate the performance of the proposed algorithm and finally, we present the conclusions in Section VII.

III. SYSTEM MODEL AND PROBLEM FORMULATION

A. NETWORK MODEL

In this paper, we considered an IoT-fog interconnection network, as shown in Fig. 1. It consists of n fog nodes and m IoT devices, denoted as $\mathbb{F} = \{f_1, f_2, \ldots, f_n\}$ and $\mathbb{D} = \{d_1, d_2, \ldots, d_m\}$, respectively. For resource allocation decisions in the network, a fog node is designated as the Fog Node Controller (FNC), and it assigns tasks to fog nodes using a many-to-one matching game, as discussed later in this article. It is assumed that each IoT device generates a single heterogeneous sized task, which is represented by the task set, $\mathbb{T} = \{t_1, t_2, \ldots, t_m\}$, where task t_m corresponds to the task generated by the IoT device d_m.

It is assumed that IoT devices have limited computational capabilities and must rely on fog node resources to complete their tasks. When a task t_m is generated, the IoT device d_m sends an offloading request to the FNC in the form of a tuple $(W_m, C_{bit}, T_m^{max})$, where W_m (in bits) represents the input task

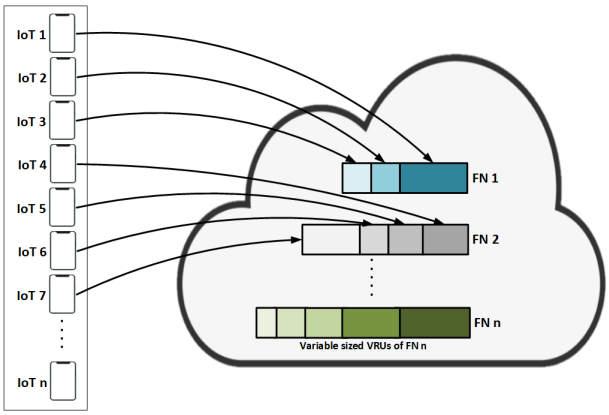


FIGURE 1. System model.

size, C_{bit} (cycles) represents the number of Central Processor Unit (CPU) cycles required to compute one bit of the task and, T_m^{max} (seconds) represents the task deadline.

When using matching theory for resource allocation, other researchers logically partition computation resources C_{f_n} of fog node f_n into homogeneous sized VRUs, whereas VRUs between different fog nodes are considered heterogeneous sized. In contrast, we present a novel concept of variable sized VRUs with fog nodes, the size of which adjusts dynamically to meet the precise resource requirements of IoT device tasks.

B. LATENCY MODEL

The latency incurred in computing IoT device tasks at the fog node is determined by:

1) TRANSMISSION DELAY

It is the time it takes to transmit a task from an IoT device to a fog node. It is assumed that an active up-link with dedicated bandwidth B_{mn}^u exists between an IoT device d_m and a fog node f_n. The task uplink rate R_{mn}^u is calculated as [28]:

$$R_{mn}^u = B_{mn}^u log_2\left(1 + \frac{p_{mn}^u g_n}{\sigma^2}\right) \tag{1}$$

where, p_{mn}^u represents the transmit power of d_m, σ represents the white noise power and g_n represents the channel gain. The channel gain varies inversely with the distance between d_m and f_n. The transmission time of a task t_m with size W_m is calculated as:

$$T_{mn}^u = \frac{W_m}{R_{mn}^u} \tag{2}$$

2) QUEUING DELAY AT THE FOG NODE

This is the amount of time a task spends in the fog node queue before it is executed. In this paper, dedicated fog node computation resources are assigned to each task, resulting in no queuing delay at the fog node.

3) COMPUTATION LATENCY AT THE FOG NODE

This is the amount of time spent at the fog node actually performing the task. Given that c_{nm} is the amount of computational resources allocated by fog node f_n for task t_m from its total free computation resource C_{f_n}, the task computation latency at f_n is calculated as:

$$T_{mn}^f = \frac{W_m C_{bit}}{c_{nm}} \tag{3}$$

4) RESULT DOWNLOAD LATENCY

It is the amount of time required to transmit the processed output from the fog node to the relevant IoT device. In this paper, we assume that the output O_m is very small in comparison to the input, so download latency is ignored [29].

Total computation latency of task t_m is the sum of all above latencies and, from Eq. 2 and 3:

$$T_{mn}^{tot} = \frac{W_m}{R_{mn}^u} + \frac{W_m C_{bit}}{c_{nm}} \tag{4}$$

C. ENERGY CONSUMPTION MODEL

Both IoT devices and fog nodes consume energy during the task offloading process in the following ways:

1) ENERGY CONSUMPTION OF IoT DEVICES

The energy consumed by the IoT device while offloading tasks to the fog node is calculated as:

$$E_{mn}^d = P_{mn}^u T_{mn}^u \tag{5}$$

2) ENERGY CONSUMPTION OF FOG NODES

The fog node consumes energy in: (1) receiving the task offloaded from the IoT device d_m and, (2) computing the received task. We know that the time it takes an IoT device to transmit a task is the same as the time it takes a fog node to receive it. If P_{mn}^r is the fog node power consumed by the fog node in receiving the offloaded task and P_n^c is the power consumed by the fog node in task computation, then the energy consumed in task computation at the fog node can be calculated as [30].:

$$E_{nm}^f = P_{mn}^r T_{mn}^u + P_n^c T_{mn}^f \tag{6}$$

3) TOTAL ENERGY CONSUMED

It is the sum of energy consumed by IoT devices and the fog node:

$$E_m = E_{mn}^d + E_{nm}^f \tag{7}$$

D. TASK OUTAGES

Outages occur when tasks are not completed by the deadline, i.e., $T_{mn}^{tot} > T_m^{max}$. In our system settings, an IoT device cannot locally process the task; thus, an IoT device task suffers outage when it is not matched with a fog node. Let x_{mn} be a binary indicator variable that indicates whether a task t_m is assigned to a fog node f_n or not as:

$$x_{mn} = \begin{cases} 1: & \text{if } t_m \text{ is assigned to} f_n \\ 0: & \text{otherwise} \end{cases} \tag{8}$$

Then the set of tasks suffering outages $\mathbb{O}$ can be formally defined as:

$$\mathbb{O} = \left\{t_m \in \mathbb{T} \mid T_{mn}^{tot} > T_m^{max} \vee x_{mn} = 0\right\} \tag{9}$$

E. MINIMUM RESOURCE REQUIREMENT OF IoT DEVICE TASKS TO REDUCE TASK OUTAGES

According to the previously stated definition, a task outage occurs when it is not completed by the deadline. Eq. 4 describes the relationship between allocated fog node computation resources and task completion time. To avoid an outage, the bare minimum computation resources c_{nm}^{min} required by a task from the fog node can be determined by reverse-calculating the fog node computation resources in Eq. 4 by equating T_{mn}^{tot} to T_m^{max} as:

$$c_{nm}^{min} = \frac{W_m C_{bit}}{(T_m^{max} - T_{mn}^{u})} \tag{10}$$

Because an IoT device has a different distance and transmission rate from each fog node, an IoT device task will require a different amount of computation resources from each fog node to complete by the deadline. The computation resource requirement in Eq. 10 accounts for both task transmission time from the IoT device to the fog node and task computation time at the fog node. Only fog nodes with $C_{f_n} \geq c_{nm}^{min}$ can complete t_m before the deadline and avoid an outage:

F. PROBLEM FORMULATION

The objective of this paper is to minimise task outages by increasing the the number of tasks accepted at fog nodes while optimising energy efficiency through an efficient resource allocation strategy based on a many-to-one matching game. If Q represents the total number of tasks accepted by fog nodes, the optimization problem can be formulated as:

Problem (P1):

$$\min\ \mathbb{O} \text{ by max } Q \text{ and min } E_m$$

$$s.t.\ T_{mn}^{tot} \leq T_m^{max} \tag{11a}$$

$$\sum_{i=1}^{n} x_{mi} = 0 \implies t_m \in \mathbb{O} \tag{11b}$$

$$\sum_{m=1}^{q_n} c_{nm} \leq C_{f_n} \tag{11c}$$

Constraint (11a) ensures that task t_m is computed within deadline. Constraint (11b) ensures that if task is not assigned to any fog node, it is defined as a task outage case. Constraint (11c) ensures that the sum of fog node computation resources allocated to q_n number of IoT devices does not exceed the total free computation resources C_{f_n}, made available by the fog node f_n to compute tasks of IoT devices.

The formulated problem is to reduce task outages while minimising system energy consumption. This is an NP hard combinatorial optimization problem that is nearly impossible to solve in polynomial time for an increasing number of IoT devices and fog nodes [31].

IV. PROPOSED SOLUTION

In this paper, we solve the formulated problem in Section III-F using matching theory, which has recently gained momentum over classical optimization problems. Matching theory solves task offloading problems by ranking individual preferences over players in the opposite set, resulting in stable matching assignments; each agent is satisfied and has no incentive to change the allocation to which it has been assigned. Matching-based resource allocation techniques reduce task outages by formulating preference profiles that maximise the number of accepted tasks at fog nodes [17], [25]. Whereas, non-matching-based techniques reduce task outages by increasing the number of accepted tasks at the fog node. They accomplish this by increasing fog node resources or lowering the individual resource demand of all tasks from fog nodes, allowing a fog node to serve more tasks with the same resources [21], [23].

In this paper, we want to use matching theory to allocate minimal resources to tasks, just enough to complete them on time, so that the fog node can accept more tasks with the same resources, reducing task outages. Fixed-sized VRUs allocate same resources to all tasks matched to a fog node and cannot provide the desired functionality. As a result, we propose and use the concept of variable-sized VRUs, which are precisely sized to the individual resource demands of tasks.Time or power efficiency, or both, can be obtained by incorporating them into the player preference profiles.

Since the size of a variable-sized VRU is defined by the individual resource demand of a task, the total number of VRUs into which a fog node's computation resources are divided will be determined by the collective resource demands of all tasks that are matched to it. This generates a unique matching challenge in which the total capacity of a fog node to accept the number of tasks is unknown until the matching process is completed. To address this issue, we propose SMRETO, a novel many-to-one matching algorithm that is a modified implementation of the Deferred Acceptance Algorithm (DAA) [19].

TABLE 2. Notations used in the paper.

$\mathbb{F}, \mathbb{D}, \mathbb{T}, \mathbb{O}$	Set of: Fog nodes, IoT devices, Tasks and Task outages
n, m	Number of: Fog nodes and IoT devices
d_m, t_m	IoT device and IoT device task
W_m, C_m, T_m^{max}	IoT device task size, CPU cycles required for IoT task and IoT task cut off time
C_n, c_{nm}	Fog node available computation resources and Fog node computation resources allocated to IoT device d_m
c_{nm}^{min}	Minimum resources required to compute task in deadline
Q, q_n	Total number of accepted tasks, and quota of fog node
B_{mn}^{u}, R_{mn}^{u}	Bandwidth between I_m and F_n and transmission rate
σ, g_n, P_{mn}^{u}	White noise power, channel gain and transmit power
P_{mn}^{r}, P_n^{c}	Fog node task receive power and task computation power
T_{mn}^{u}, T_{mn}^{f}	Task uplink time and task computation time at fog node
T_{mn}^{tot}, E_{mn}^{d}	Task total time and energy consumed in IoT device
E_{nm}^{f}, E_m	Energy consume in fog node and task total energy
$\succ_{f_n}, \succ_{t_m}$	f_n preference profile and Task preference profile
f_n^A, t_m^A	f_n association set and d_m association set

A. MATCHING CONCEPTS

Formally, matching game is expressed as per definition 1:

Definition 1: Let $\mathbb{F}$ and $\mathbb{T}$ be two sets of fog nodes and IoT device tasks, respectively. A matching assignment defined over ($\mathbb{F}$, $\mathbb{T}$) has two sets of preference relations $\succ_F$ and $\succ_T$ that allows each player ($f_n \in \mathbb{F}$) to indicate preference over all players ($t_m \in \mathbb{T}$) in the opposite set, and vice versa.

Definition 2: Matching assignment is based on a mapping function λ, such that:

$$\lambda(t_m) \subseteq \mathbb{F} \text{ and } |\lambda(t_m)| \leq 1 \tag{12a}$$

$$\lambda(f_n) \subseteq \mathbb{T} \text{ and } |\lambda(f_n)| \leq q_n$$

$$\text{s.t} \sum_{1}^{q_n} c_{nm} \leq C_{f_n} \tag{12b}$$

$$f_n \in \lambda(t_m) \iff t_m \in \lambda(f_n) \tag{12c}$$

Condition (12a) shows that an IoT device task t_m can only have one match with single fog node only. The condition (12b) states that a fog node can have q_n matches with q_n IoT device tasks only if the sum of computation resources allotted to these IoT device tasks does not exceed the fog node's total free computational resources C_{f_n} (explained later in this article). Condition (12c) implies that t_m is matched to f_n *iff* f_n is matched to t_m.

B. ASSOCIATION BETWEEN FOG NODE AND IoT DEVICE TASKS

We know from Eq. 10, that a fog node must have computation resources greater than c_{nm}^{max} in order to complete a task on time. Therefore, when FNC receives an offloading request, it first calculates the task transmission time (using channel state information, periodically provided to FNC by fog nodes) and checks the available computation resources of all fog nodes, shortlisting fog nodes satisfying these conditions in the task association set t_m^A. The fog node association set f_n^A is populated on the basis that if a fog node is associated with an IoT device task, then that IoT device task is also associated with that fog node. A fog node's f_n free computational resources C_{f_n} are matched only to those tasks, where $t_m \in f_n^A$.

C. PREFERENCE PROFILE OF PLAYERS

The preference profile of each player is used to rank the players in the opposite set. It defines the order in which players from the opposite set satisfy the player's objective function. In our work, the preference profile of both IoT device tasks and fog nodes is set to achieve energy efficiency. Since energy consumption of an IoT device is due to task offloading, which is dependent on transmission rate, therefore, a fog node with a higher transmission rate is preferred over one with a lower transmission rate, i.e.,

$$f_n \succ_{t_m} f_{n'} \iff R_{mn}^u > R_{mn'}^u \tag{13}$$

For a fog node, preference profile is defined as:

$$t_m \succ_{f_n} t_{m'} \iff E_{nm}^f < E_{nm'}^f \tag{14}$$

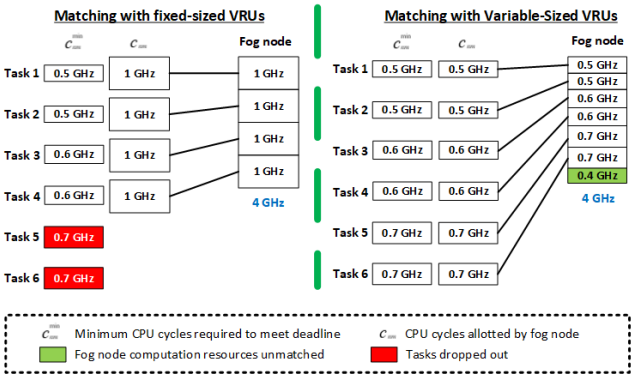


FIGURE 2. Matching with fixed and variable sized VRUs.

D. VARIABLE SIZED VRUs AND QUOTA OF FOG NODES

The quota represents a fog node's capacity to accept a specific number of tasks and it is determined by the number of VRUs into which fog node computation resources are logically partitioned. In our work, we introduced the concept of variable-sized VRUs that are tailored to the heterogeneous resource requirements of tasks. The number of such VRUs that will be formed from the computation resources of a fog node cannot be predicted in advance. Hence, once the matching process starts, we must treat a fog node's quota as one until it receives a match. If a fog node receives a match and has some free computation resources, its quota must be treated as two. Similarly, the fog node's quota increases until all of its computation resources are depleted. We will know the quota of each fog node once the matching process is completed.

Consider Fig. 2, which depicts a matching process with fixed-sized VRUs on the left and a matching process with variable-sized VRUs on the right. There are six tasks that require [0.5, 0.5, 0.6, 0.6, 0.7, 0.7] GHz CPU cycles from the only fog node available to complete before the deadline. The fog node has 4 GHz to serve all tasks, which are logically partitioned into four fixed sized VRUs of 1 GHz each. Since the fog node has a quota of four, it matches with four tasks and allocates 1 GHz of computation resources to each, while two tasks suffer outage.

Variable-sized VRUs are precisely sized to the resource requirements of IoT device tasks, so they will match all six tasks while saving 0.4 GHz of the fog node's computation resources. The fog node can use these resources to accept more tasks. In Fig. 2. If we are in the middle of the matching process, the fog node's quota will be seven, and if the matching process is finished, the fog node's quota will be six.

E. MODIFIED IMPLEMENTATION OF DAA

In our work, two algorithms are used in FNC to match fog node computation resources to tasks:

1) INPUT GENERATION FOR MATCHING ALGORITHM

Calculates the transmission rate between IoT devices and fog nodes to determine the minimum resource requirement

Algorithm 1 Input Generation for Matching Algorithm

Input: W_m, C_m, T_m^{max} and CSI
Output: c_{nm}^{min}, f_n^A, t_m^A, $\succ_{f_n}$ and $\succ_{t_m}$
for $\forall t$ **do**
  for $\forall f$ **do**
    Calculate transmission rate R_{mn}^u
    Calculate resource requirement c_{nm}^{min}
    Generate association sets
  end
end
for $\forall t \in f_n^A$ **do**
  Calculate fog nodes preference profile $\succ_{f_n}$
end
for $\forall f \in t_m^A$ **do**
  Calculate IoT device task preference profile $\succ_{t_m}$
end

c_{nm}^{min} for each task t_m from each fog node. Generates association sets and preference profiles using the steps shown in Algorithm (1).

2) PROPOSED SMRETO ALGORITHM

The proposed SMRETO is a modified DAA implementation that uses variable-sized VRUs with fog nodes. SMRETO computes fog node quota on the go, based on the resource requirements of the held task proposal and the fog node's remaining computation resources. The steps involved are shown in Algorithm (2). We also use Fig. 3 to explain how the proposed SMRETO algorithm works with one fog node and eight tasks.

The quota of all fog nodes is set to one. IoT device tasks propose the fog node in the order defined by their preference profile $\succ_{t_m}$. For the first proposal, if the fog node has more computation resources than the minimum resource requirements of the proposing task, the proposal is held. For the remaining proposals, SMRETO compares the preference of the new proposal to the preference of the held proposals. If the preference for the new proposal is lower than the preference for the held proposals, SMRETO determines whether the fog node has enough remaining resources to serve both the new and held proposals.

If the fog node has the required resources, all proposals are retained; otherwise, the new proposal is rejected. All tasks in f_n^A that have not yet proposed to f_n, but are lower in the preference profile $\succ_{f_n}$ than the rejected proposal are removed from f_n^A and f_n is removed from $t_{m'}^A$. (This step ensures that no lower priority tasks are accepted under any circumstances if a higher priority task is rejected.) In case 1 of Fig. 3, when task 7 proposes fog node, regardless of its preference over the held proposals, the proposal of task 7 is held because fog node has enough resources to compute task 7 with already held proposals.

If the new proposal has a higher preference than at least one of the held proposals, SMRETO determines whether the fog node can serve all of the proposals or not. If yes, all proposals are retained; otherwise, SMRETO evaluates the serviceability of new and held proposals with a lower preference than the received proposal. It starts rejecting the held proposals in reverse order of $\succ_{f_n}$ until the resource requirements of the remaining proposals can be met within C_{f_n}. Even after rejecting all held proposals with lower preference than the proposing task, the fog node may lack the resources to accept that task. In that case, the proposing task is also rejected.

In case 2 of Fig. 3, when task 5 proposes the fog node, its priority order with the fog is higher than held proposals of tasks 1, 7, and 6. Task 5 requires 0.9 GHz resources, but the fog node only has 0.3 GHz of free computation resources. Thus, the fog ode cannot retain the proposal of task 5, and if it rejects the proposal of task 5, it must also reject the proposals of tasks 1, 7, and 6. To avoid large scale rejections, the fog node first rejects the least preferred of the fog node's held proposals, task 6, and then determines whether it can now serve the remaining held proposals. The fog node still lacks the resources required to compute the remaining tasks. The fog node then rejects the least preferred held proposal of task 7 and finds that it can now serve the remaining tasks. The fog node retains proposals of tasks 5 and 1 and removes tasks 7, 6, and 8 from its association set. The fog node updates its quota information.

The rejected IoT device task proposes to the next fog node in its preference profile. The process is repeated until all tasks are either matched or they have exhausted the options of fog nodes in their association sets. The proposed SMRETO starts matching with a fog node quota value of one. With each new proposal, SMRETO recalculates a fog node's quota as: (a) If the fog node keeps the new proposal along with the held proposals, the fog node's quota is increased by one; (b) If the new proposal is accepted but one of the held proposals is rejected, fog node's quota remains unchanged, and (c) If more than one held proposal is rejected, the fog node's quota is reduced by one less than the number of held proposals rejected. The number of matches with a fog node will be the final quota for the fog node.

F. RESOURCE SCALING TO IMPROVE ENERGY OR TIME EFFICIENCY

The proposed SMRETO ensures that all tasks are served using the least amount of fog node computation resources. This reduces the amount of computation resources needed to compute the tasks, and a fog node can serve more tasks with same resources, resulting in fewer task outages. SMRETO on the other hand, achieves energy or time efficiency by formulating players preference profiles based on their energy and time consumption. Because SMRETO always assigns the bare minimum of computation resources to tasks, one could argue that in a low workload scenario, it under utilizes available computation resources that could be used to further improve energy and time efficiency. This is because task completion time and energy consumption are inversely

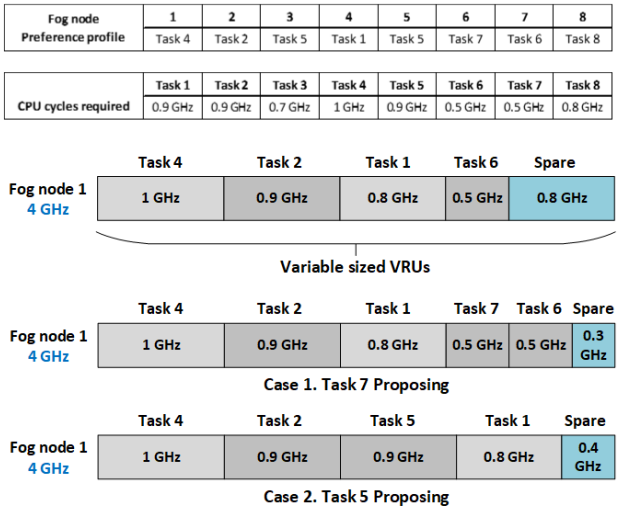


FIGURE 3. Working of SMRETO Algorithm.

proportional to the amount of computation resources used to compute the task.

If more energy and time efficiency are essentially required, the problem is easily solved by simply scaling up the computational resources already committed to the tasks in the matching process from the fog node's remaining computation resources. To scale up, we can use all or part of the fog node's remaining computation resources. Scaling up allows for the addition of more resources in proportion to those already committed, i.e., the greater the value of committed resources, the greater the value added to it. If the total free computation resources available with fog node f_n were C_{f_n} and f_n matched $C_{f_{n'}}$ resources in the matching process, the resources c_{nm} that will be issued to task t_m in addition to already committed resources c_{nm}^{min} can be calculated as:

$$c_{nm} = c_{nm}^{min}(C_{f_n}/C_{f_{n'}}) \tag{15}$$

The total computation resources for task t_m would be $(c_{nm} + c_{nm}^{min})$.

V. THEORETICAL ANALYSIS

This section includes a theoretical analysis of the proposed SMRETO's stability, a description of how SMRETO is a modified implementation of DAA, and a complexity calculation for the SMRETO algorithm.

A. STABILITY ANALYSIS

Every matching algorithm's goal is to find a stable offloading decision while taking into account the individual preferences ranking over the players in the opposite set. The key concept of stability in matching theory is defined as follows.

Definition 3: A matching function λ is individually rational *iff* there does not exists an agent $a \in \mathbb{T}U\mathbb{F}$ such that $\phi \succ_a \lambda(a)$, i.e., agent a prefers to remain unmatched in comparison to its current match $\lambda(a)$.

This definition implies that the partner found through the matching process must be acceptable. In our paper settings, fog nodes with computation resources less than the task's minimum computation resource requirements are unacceptable, and we ensured that no match is made with an unacceptable fog node by defining the association set and matching within the bounds of the association set. As a result, SMRETO's matching assignments are individually rational.

Definition 4 (Blocking Pair): A matching function λ is said to be blocked by a pair of agents (f_n, t_m) *iff* $f_n \prec_{t_m} f_{n'}$, $t_m \prec_{f_n} t_{m'}$, but $t_m \notin \lambda(f_n)$, $t_{m'} \in \lambda(f_n)$ and similarly $f_n \notin \lambda(t_m)$, $f_{n'} \in \lambda(t_m)$, i.e., A pair (f_n, t_m) blocks assignment λ when they are not matched with each under current λ but they prefer to be matched with each other.

When a proposal is rejected in proposed SMRETO, all tasks in f_n^A that have not yet proposed to f_n, but are lower in the preference profile $\succ_{f_n}$ than the rejected proposal are also deleted. This ensures that there are no blocking pairs in SMRETO's matching assignments.

Definition 5 (Stable Matching): A matching function λ is said to be stable *iff* it is individually rational and is not blocked by any pair of agents.

Algorithm 2 Proposed Matching Algorithm

Input: C_n, c_{nm}^{min} f_n^A, t_m^A, $\succ_{f_n}$ and $\succ_{t_m}$
Output: Matching assignment: λ
For better understanding, lets name proposing task as t^p, proposals held as t^h and rejected proposal as t^r
while $(\forall t,\ \lambda(t_m) \subseteq \mathbb{F} \ \vee \ t_m^A = \phi)$ **do**
  for $\forall t$ **do**
    Propose top f_n in $\succ_{t_m}$.
    if *(first proposal with* f_n*)* **then**
      Hold t^p
    else if $(t^h \succ_{f_n} t^p)$ **then**
      if $(\forall proposals \sum c_{nm}^{min} \leq C_{nm})$ **then**
        Hold t^p
      else
        Reject t^p and $\forall t_{m'}$ s.t $t^p \succ_{f_n} t_{m'}$ delete $t_{m'}$ from f_n^A and f_n from $t_{m'}^A$
      end
    else
      if $(\forall proposals \sum c_{nm}^{min} \leq C_{nm})$ **then**
        Hold t^p along with other t^h
        Reject t^p and $\forall t_{m'}$ s.t $t^p \succ_{f_n} t_{m'}$ delete $t_{m'}$ from f_n^A and f_n from $t_{m'}^A$
      else
        for t^p *and* $\forall t^{h'}$ *s.t* $t^p \succ_{f_n} t^{h'}$ **do**
          Keep rejecting lowest proposals in $\succ_{f_n}$ until $\sum c_{nm}^{min} \leq C_{nm}$
          $\forall t_{m'}$ s.t $t^r \succ_{f_n} t_{m'}$ delete $t_{m'}$ from f_n^A and f_n from $t_{m'}^A$
        end
      end
    end
  end
end

TABLE 3. Simulation settings.

Parameter	Value
Number of fog nodes	5
Number of IoT devices	200-1000
Fog node computation resources	U[4,6] GHz
Fog node computation power	U[0.35,0.55] W
Input task size	U[500,600] Kb
Computational demand of a task	U[500,750] million cycles
Task deadline	U[20,30] s
Dedicated bandwidth for up-link	10×10^6 Hz
IoT transmission power	U[0.1,1] W
Fog node received power	U[1,2] W
White noise power	10×10^{-10} W

Because SMRETO meets both of the necessary conditions for stable matching assignments, we can safely conclude that SMRETO makes stable offloading decisions.

B. SMRETO - MODIFIED IMPLEMENTATION OF DAA

The proposed SMRETO works similarly to DAA, with the exception that DAA matches with a known value of fog node quota, whereas SMRETO does not know what the value of fog node quota will be and starts matching with a fog node quota value of one. The quota is incriminated as long as the fog node has enough resources to serve all tasks simultaneously. When the fog node's capacity is exhausted, the selection of proposals begins in the order of their preference. The number of matches with a fog node will be the final quota for the fog node. When we run DAA with the fog node quota finalised by SMRETO, we get the same stable matching result.

C. COMPLEXITY ANALYSIS

The overall time complexity of the proposed SMRETO depends on the complexity of the matching algorithm, which is given by $O(m \times n)$. The overall complexity of the algorithm is thus polynomial in time.

VI. PERFORMANCE EVALUATION

In this section, we present simulation results for various network configurations to compare the performance and computational efficiency of our proposed algorithm to other algorithms in the literature.

A. SIMULATION SETUP

We developed a fog network simulation setup in MATLAB and implemented the task offloading scenario. The value of key simulation parameters is displayed in Table 3. The fog network consists of 5 fog nodes spread across an area of 100 m $\times$ 100 m. The computation resources (cycles/s) and the computational power (W) of fog nodes are considered to be heterogeneous and uniformly distributed in the range of 4-6 GHz and 0.35-0.55 W respectively. The number of IoT devices are considered to be in the range of 200 to 1000, with a difference of 100 devices between simulation iterations. Each IoT device generates a single task with input task size (bits), CPU cycles required to complete the task (cyles/s) and, task deadline (s) uniformly distributed in the range of 300-600 Kb, 500-750 million cycles and, 20-30 s.

Each IoT device has an active up-link with a dedicated bandwidth of 10 MB with each fog node, ensuring no channel access wait time. Considering PCS-1900 GSM band, the free space path loss in dB between an IoT device d_m and fog node f_n is calculated as: $PL_{m,n} = 38.02 + 20 \log d_{m,n}$ [17]. The channel gain g_n is then calculated as: $g_n = 10^{-PL_{m,n}/10}$. The communication channel is assumed to be noisy, with noise power $\sigma^2 = 10^{-10}$.

B. BASELINE ALGORITHMS

We compare the performance of proposed SMRETO with three baseline algorithms: (1) Swain et al. [17] (referred to as METO), (2) Chittaranjan et al. [25] (referred to as SPATO) and, (3) Chiti et al. [18] (referred to as ME).

The three schemes model their resource allocation strategy using a matching game. The baseline schemes METO and SPATO aim to reduce task outages with energy optimization, whereas, ME aims to achieve time efficiency. For simulation purposes, the caching delay in ME is ignored, and the preferences from IoT devices and FNs are based on minimum completion and deadline, respectively.

C. RESULTS AND ANALYSIS

The primary performance metric used to compare proposed SMRETO to baseline schemes is the reduction in the number of task outages under different workload scenarios. The auxiliary performance metrics are the percentage of total available computation resources used, total system energy consumed, mean energy per task measured in Joules, and task execution time measured in seconds.

The proposed SMRETO allocates the bare minimum computation resources to each task in order to conserve fog node resources and serve more tasks with the same fog node resources. If some fog node computation resources remain unused in low workload scenarios, we can scale up the allocated resources from unused fog node computation resources to improve energy efficiency even further. We show the results of task outages and resources utilisation percentages without increasing the resources already assigned to tasks. In contrast, the results of energy consumption and latency are shown after scaling up the allocated resources from the fog node's remaining computation resources.

1) TASK OUTAGES

Fig 4 shows the number of task outages experienced by all of the schemes under consideration. The results clearly show that the proposed scheme SMRETO outperforms other competing algorithms in terms of task outages. Other baseline schemes can only match SMRETO's performance in low workload scenarios (when the number of tasks is low),

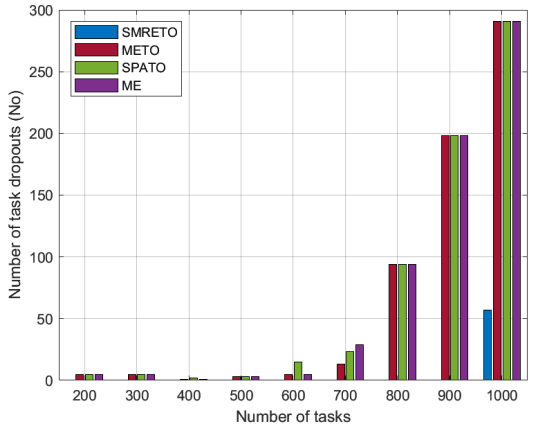


FIGURE 4. Task outages.

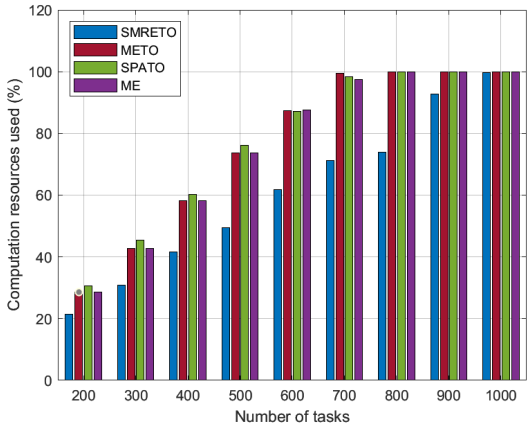


FIGURE 5. Utilization of system resources.

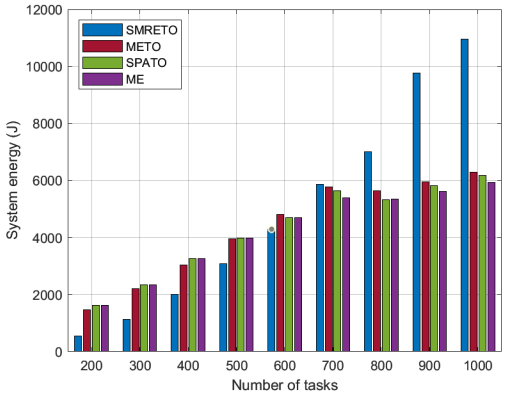


FIGURE 6. System energy consumed.

whereas SMRETO performs best in high workload scenarios. The key to such performance is allocating resources for task deadlines in order to maximise the number of accepted tasks at the fog nodes. This is made possible by variable-sized VRUs with fog nodes that can precisely size to the resource requirements of the matched IoT device tasks. The results show that with variable-sized VRUs, task outages occur only when fog node resources are completely depleted in high workload scenarios.

To allocate resources with fixed-sized VRUs, we must first define the total number of VRUs into which fog node computation resources are logically distributed. After a large number of simulations with our system parameter settings, we found that the total number of VRUs in the system should be between 650 and 750 for the best task latency results for the baseline schemes under consideration. Among the baseline schemes, both METO and SPATO incorporate task deadlines with energy minimization when formulating the fog node preference profile. This enables these schemes to prioritise tasks with short deadlines, increasing the number of accepted tasks at fog nodes and, as a result, reducing task outages. ME, on the other hand, does not consider task deadlines and thus has a higher number of task outages. With baseline schemes that use fixed-sized VRUs, all tasks that exceed the number of VRUs available in the system will experience task outage.

2) FOG NODE RESOURCE UTILIZATION

Fig 5 shows the percentage of total fog node resources used to serve the accepted tasks. In comparison to the baseline schemes, the proposed SMRETO uses the least amount of the fog node's computation resources to compute the same number of tasks. In contrast, other baseline schemes use more resources to serve a smaller number of tasks. When these results are compared to those in Fig 4, they show that in a high workload scenario, some fog node resources remain unused, while the baseline schemes experience task outages. This inability to fully utilise available resources stems from the inherent limitation associated with fixed-size VRUs, which are always under-sized or over-sized in comparison to IoT device resource requirements. Only over-sized VRUs will be matched to an IoT device task to avoid task outage. When the system is overloaded, there will always be some under-sized VRUs and some IoT device tasks with high resource demands, both of which will remain unmatched. The solution is to use flexible sized VRUs, as SMRETO has done.

3) ENERGY AND TASK LATENCY

We know from Eq. (3) that task computation time at the fog node is inversely proportional to the amount of fog node resources committed to compute the IoT device task, and we also know from Eq. (6) that task computation time is directly proportional to the energy consumed at the fog node for task computation. As a result, the amount of computation resources provided by fog nodes to IoT device tasks has a significant impact on time and energy efficiency. For this reason, we first reduce task outages by allocating the bare minimum of fog node computation resources, and then we scale up the allocated resources from fog node free computation resources to improve time and energy efficiency.

Fig 6 shows the system energy consumed in computing all accepted tasks, Fig 7 shows the corresponding mean energy consumed in computing a single task, and Fig 8 shows the mean task latency. The three figures show the results when the proposed SMRETO fully utilises the fog node computation resources. These figures show that when the workload is low

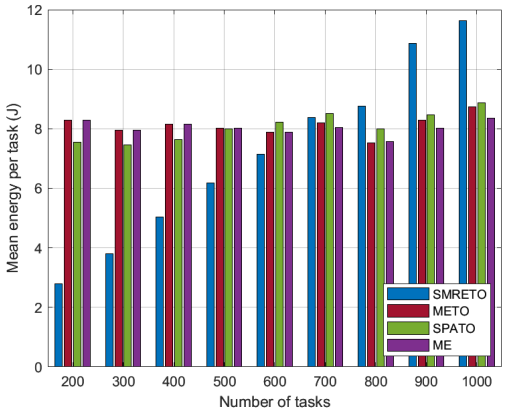


FIGURE 7. Mean energy per accepted task.

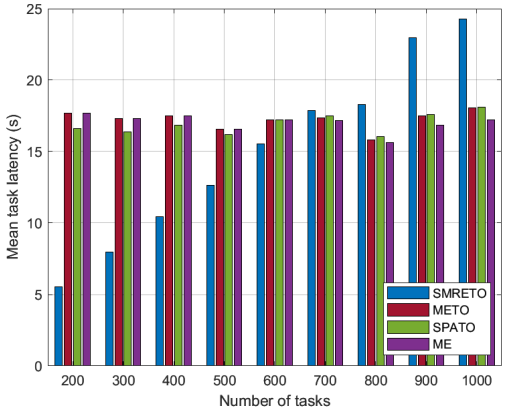


FIGURE 8. Mean task latency.

to medium, SMRETO outperforms all baseline schemes. This is because variable-sized VRUs in the proposed SMRETO can dynamically adjust their sizes and use all available computation resources. However, when the workload is high, even the proposed SMRETO experiences task outages. In this case, SMRETO consumes all fog node computation resources to avoid task outages and has no spare computation resources to improve energy efficiency. In such cases, the proposed SMRETO algorithm trades off energy and time efficiency for a lower number of task outages.

VII. CONCLUSION

In this paper, we have proposed an IoT device to fog node task offloading algorithm to minimize the number of task outages and reduce the system energy consumption. To achieve these goals, the proposed technique uses variable sized computing resources on the fog nodes (known as VRUs) and IoT task requirements are based on the task deadline. The proposed algorithm utilizes a many-to-one matching algorithm to allocate IoT tasks to the variable sized VRUs. The preference profile of IoT tasks and fog computing resources are developed to ensure reduction of system energy consumption. The unmatched resources at the fog nodes are also utilized towards computing of allocated tasks. Simulation results highlight the advantages of the proposed algorithm in terms of task outages and system energy consumption.

REFERENCES

[1] K. B. Letaief, Y. Shi, J. Lu, and J. Lu, "Edge artificial intelligence for 6G: Vision, enabling technologies, and applications," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 5–36, Jan. 2022.

[2] T. Yang, M. Qin, N. Cheng, W. Xu, and L. Zhao, "Liquid software-based edge intelligence for future 6G networks," *IEEE Netw.*, vol. 36, no. 1, pp. 69–75, Jan. 2022.

[3] U. M. Malik, M. A. Javed, S. Zeadally, and S. U. Islam, "Energy-efficient fog computing for 6G-enabled massive IoT: Recent trends and future opportunities," *IEEE Internet Things J.*, vol. 9, no. 16, pp. 14572–14594, Aug. 2022.

[4] X. Shen, J. Gao, W. Wu, M. Li, C. Zhou, and W. Zhuang, "Holistic network virtualization and pervasive network intelligence for 6G," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 1–30, 1st Quart., 2022.

[5] M. A. Javed, T. N. Nguyen, J. Mirza, J. Ahmed, and B. Ali, "Reliable communications for cybertwin-driven 6G IoVs using intelligent reflecting surfaces," *IEEE Trans. Ind. Informat.*, vol. 18, no. 11, pp. 7454–7462, Nov. 2022.

[6] B. Mao, F. Tang, Y. Kawamoto, and N. Kato, "AI models for green communications towards 6G," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 210–247, 1st Quart., 2022.

[7] J. Ahmed, T. N. Nguyen, B. Ali, A. Javed, and J. Mirza, "On the physical layer security of federated learning based IoMT networks," *IEEE J. Biomed. Health Informat.*, early access, May 10, 2022, doi: 10.1109/JBHI.2022.3173947.

[8] M. A. Javed, M. Z. Khan, U. Zafar, M. F. Siddiqui, R. Badar, B. M. Lee, and F. Ahmad, "ODPV: An efficient protocol to mitigate data integrity attacks in intelligent transport systems," *IEEE Access*, vol. 8, pp. 114733–114740, 2020.

[9] M. A. Javed, S. Zeadally, and E. B. Hamida, "Data analytics for cooperative intelligent transport systems," *Veh. Commun.*, vol. 15, pp. 63–72, Jan. 2019.

[10] A. H. Sodhro, M. S. Obaidat, S. Pirbhulal, G. H. Sodhro, N. Zahid, and A. Rawat, "A novel energy optimization approach for artificial intelligence-enabled massive Internet of Things," in *Proc. Int. Symp. Perform. Eval. Comput. Telecommun. Syst. (SPECTS)*, Jul. 2019, pp. 1–6.

[11] L. Bariah, L. Mohjazi, S. Muhaidat, P. C. Sofotasios, G. K. Kurt, H. Yanikomeroglu, and O. A. Dobre, "A prospective look: Key enabling technologies, applications and open research topics in 6G networks," *IEEE Access*, vol. 8, pp. 174792–174820, 2020.

[12] Q. Qi, X. Chen, and D. W. K. Ng, "Robust beamforming for NOMA-based cellular massive IoT with SWIPT," *IEEE Trans. Signal Process.*, vol. 68, pp. 211–224, 2020.

[13] U. Farooq, M. W. Shabir, M. A. Javed, and M. Imran, "Intelligent energy prediction techniques for fog computing networks," *Appl. Soft Comput.*, vol. 111, Nov. 2021, Art. no. 107682. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494621006037

[14] A. H. Sodhro, G. H. Sodhro, M. Guizani, S. Pirbhulal, and A. Boukerche, "AI-enabled reliable channel modeling architecture for fog computing vehicular networks," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 14–21, Apr. 2020.

[15] M. Rahim, S. Ali, A. N. Alvi, M. A. Javed, M. Imran, M. A. Azad, and D. Chen, "An intelligent content caching protocol for connected vehicles," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 4, p. e4231, Apr. 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.4231

[16] N. C. Luong, Y. Jiao, P. Wang, D. Niyato, D. I. Kim, and Z. Han, "A machine-learning-based auction for resource trading in fog computing," *IEEE Commun. Mag.*, vol. 58, no. 3, pp. 82–88, Mar. 2020.

[17] C. Swain, M. N. Sahoo, A. Satpathy, K. Muhammad, S. Bakshi, J. J. P. C. Rodrigues, and V. H. C. de Albuquerque, "METO: Matching-theory-based efficient task offloading in IoT-fog interconnection networks," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12705–12715, Aug. 2021.

[18] F. Chiti, R. Fantacci, and B. Picano, "A matching theory framework for tasks offloading in fog computing for IoT systems," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 5089–5096, Dec. 2018.

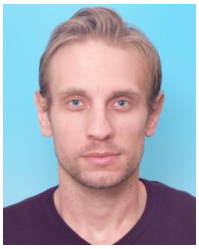
[19] A. E. Roth, "Deferred acceptance algorithms: History, theory, practice, and open questions," Nat. Bur. Econ. Res., Cambridge, MA, USA, Work. Paper 13225, Jul. 2007.

[20] Y.-L. Jiang, Y.-S. Chen, S.-W. Yang, and C.-H. Wu, "Energy-efficient task offloading for time-sensitive applications in fog computing," *IEEE Syst. J.*, vol. 13, no. 3, pp. 2930–2941, Sep. 2019.
[21] Y. Wu, Y. Wang, Y. Wei, and S. Leng, "Intelligent deployment of dedicated servers: Rebalancing the computing resource in IoT," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, Apr. 2020, pp. 1–6.
[22] B. Omoniwa, R. Hussain, M. Adil, A. Shakeel, A. K. Tahir, Q. U. Hasan, and S. A. Malik, "An optimal relay scheme for outage minimization in fog-based Internet-of-Things (IoT) networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3044–3054, Apr. 2019.
[23] R. A. C. da Silva and N. L. S. da Fonseca, "Location of fog nodes for reduction of energy consumption of end-user devices," *IEEE Trans. Green Commun. Netw.*, vol. 4, no. 2, pp. 593–605, Jun. 2020.
[24] Y. Kyung, "Prioritized task distribution considering opportunistic fog computing nodes," *Sensors*, vol. 21, no. 8, p. 2635, Apr. 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/8/2635
[25] C. Swain, M. N. Sahoo, and A. Satpathy, "SPATO: A Student project allocation based task offloading in IoT-fog systems," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2021, pp. 1–6.
[26] H. Wu, J. Zhang, Z. Cai, Q. Ni, T. Zhou, J. Yu, H. Chen, and F. Liu, "Resolving multitask competition for constrained resources in dispersed computing: A bilateral matching game," *IEEE Internet Things J.*, vol. 8, no. 23, pp. 16972–16983, Dec. 2021.
[27] C. Swain, M. N. Sahoo, and A. Satpathy, "LETO: An efficient load balanced strategy for task offloading in IoT-fog systems," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Sep. 2021, pp. 459–464.
[28] R. Basir, S. Qaisar, M. Ali, and M. Naeem, "Cloudlet selection in cache-enabled fog networks for latency sensitive IoT applications," *IEEE Access*, vol. 9, pp. 93224–93236, 2021.
[29] X. Lyu, H. Tian, P. Zhang, and C. Sengul, "Multiuser joint task offloading and resource optimization in proximate clouds," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3435–3447, Apr. 2017.
[30] F. Chiti, R. Fantacci, and B. Picano, "A matching game for tasks offloading in integrated edge-fog computing systems," *Trans. Emerg. Telecommun. Technol.*, vol. 31, no. 2, p. e3718, Feb. 2020, doi: 10.1002/ETT.3718.
[31] D. Bertsimas and J. N. Tsitsiklis, *Introduction to Linear Optimization*, vol. 6. Belmont, MA, USA: Athena Scientific 1997.

USMAN MAHMOOD MALIK received the B.S. degree in electrical (telecom) engineering and the M.S. degree in system engineering from the National University of Science and Technology (NUST), Pakistan, in 2000 and 2016, respectively. He is currently pursuing the Ph.D. degree in electrical engineering with COMSATS University Islamabad, Pakistan. He is a Faculty Member with the Computer Software Engineering Department, NUST. His research interests include modeling and system designing, the vehicular and Internet of Things (IoT) networks, energy efficiency, and wireless power transfer.

MUHAMMAD AWAIS JAVED (Senior Member, IEEE) received the B.Sc. degree in electrical engineering from the University of Engineering and Technology Lahore, Pakistan, in August 2008, and the Ph.D. degree in electrical engineering from The University of Newcastle, Australia, in February 2015. From July 2015 to June 2016, he was a Postdoctoral Research Scientist at the Qatar Mobility Innovations Center (QMIC) on SafeITS project. He is currently working as an Associate Professor with COMSATS University Islamabad, Pakistan. His research interests include intelligent transport systems, vehicular networks, protocol design for emerging wireless technologies, and the Internet of Things.

JAROSLAV FRNDA (Senior Member, IEEE) was born in Slovakia, in 1989. He received the M.Sc. and Ph.D. degrees from the Department of Telecommunications, VSB-Technical University of Ostrava, Czechia, in 2013 and 2018, respectively. He is an Assistant Professor with the University of Zilina, Slovakia. He has authored and coauthored 24 SCI-E and nine ESCI papers in WoS. His research interests include quality of multimedia services in IP networks, data analysis, and machine learning algorithms.



JAN NEDOMA (Senior Member, IEEE) is currently an Associate Professor and the Head of the Optoelectronics Laboratory with the Department of Telecommunications, Faculty of Electrical Engineering and Computer Science, Technical University of Ostrava. He is a member of the scientific council, a member of doctoral, habilitation, and professors' committees, and a Guarantor of bachelor's study programs at Faculty of Electrical Engineering and Computer Science, Technical University of Ostrava. During his scientific career, he was the Leader or a co-investigator of more than 25 projects and has more than 175 journal articles and conference papers in his research areas. He holds ten valid Czech patents. His research interests include optical communications, optical atmospheric communications, optoelectronics, optical measurements, measurements in telecommunication technology, signal processing, fiber-optic sensors, and biomedical engineering.