IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 23 July 2024, accepted 27 August 2024, date of publication 30 August 2024, date of current version 12 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3452588

RESEARCH ARTICLE

On the Partial Offloading to Multiple Helpers-Based Task Offloading for IoT Networks

USMAN MAHMOOD MALIK[1], MUHAMMAD AWAIS JAVED[2], (Senior Member, IEEE), ABDULAZIZ ALMOHIMEED[3], MOHAMMED ALKHATHAMI[4], AND ABEER ALMUJALLI[4]

[1]Department of Computer Software Engineering, National University of Science and Technology (NUST), Islamabad 44000, Pakistan
[2]Department of Electrical and Computer Engineering, COMSATS University Islamabad (CUI), Islamabad 45550, Pakistan
[3]Computer Science Department, College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh 11432, Saudi Arabia
[4]Information Systems Department, College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh 11432, Saudi Arabia

Corresponding author: Mohammed Alkhathami (maalkhathami@imamu.edu.sa)

This work was supported and funded by the Deanship of Scientific Research at Imam Mohammad Ibn Saud Islamic University (IMSIU) (grant number IMSIU-RP23054).

ABSTRACT Ubiquitous computing is a key component of future wireless communications and Internet of Things (IoT) applications. By using distributed fog computing, data obtained from sensors can be timely processed and different tasks can be computed efficiently. As a result, IoT applications can make timely decisions and improve application reliability. In this paper, we consider a Partial Offloading to Multiple Helper (POMH) scenario where IoT device divide their tasks into subtasks that can be computed in a parallel manner. We propose a modified many-to-many matching-based task offloading algorithm to reduce the task latency in a POMH scenario. To overcome the externalities due to the dynamic preference profile of fog nodes, an intelligent proposal-based scheme is also introduced. The performance analysis of the proposed techniques shows that they provide quicker task computation as compared to different techniques available in the literature.

INDEX TERMS Internet of Things, computing, fog networks.

I. INTRODUCTION

Internet of Things (IoT) technologies along with wireless communication systems, data communications, Artificial Intelligence (AI), and learning techniques have enabled efficient machine-to-machine communications. Because of these advancements, our lifestyle has evolved and now, we desire to access all information and control all devices online, via a single device for which, more and more devices are getting connected to the Internet at a very rapid pace [1], [2], [3], [4].

For a long, cloud computing has been at the centre stage to serve IoT technologies, but cloud computing being a centralized solution, will be inefficient to meet extreme low-latency requirements of future applications. In comparison, fog computing is the decentralized solution, that brings computing and storage resources within proximity of the IoT devices [5], [6]. This enables the devices to offload their computation tasks and cache data onto these fog nodes with improved latency and low energy requirements [7], [8].

In fog computing, the challenges related to computational offloading and resource allocation have been under the spotlight for several years. Their research is mostly focused on the objective functions of improving task latency or achieving energy efficiency. These objectives are achieved, mostly by managing the offloading technique through tailored network policies. These offloading techniques carefully select the size of offloading task and the location where to offload it, that best suits achieving its desired objectives. When taking the offloading decision, the whole task can be considered as an entity or it may be divided into a variable number of sub-tasks according to the offloading technique, also called binary offloading problem and partial offloading problem, respectively.

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

Partial offloading is a complex process, based on multiple interlinked factors, such as the user's class of service, link quality, and available computation and power resources [9]. Partial offloading can be done only with that data/ task/ application, which can be decomposed into smaller parts, e.g., face recognition and image processing [10]. If we consider a modular application, whose modules can be processed individually at different devices, then the job of offloading technique is to decide which module to execute locally and which one to offload. and most importantly, where to offload the module. Partial offloading can be further classified according to the number of helping devices contributing to completing the task. If the sub-tasks are offloaded to a single device, then it is called Partial Offloading with Single Helper (POSH) and when the sub-tasks use multiple helper devices for the execution of the tasks, then it is called Partial Offloading with Many Helpers (POMH).

In POMH, there are multiple partitions of a task. In most cases, one of these sub-tasks is processed on the device itself locally whereas the rest of the subtasks are computed at multiple helper nodes/ devices. These sub-tasks are processed in parallel in these devices, which greatly reduces the task latency. The offloading decision in POMH depends upon multiple interlinked factors, which need to be balanced such that, the desired objective functions are met with optimal utilization of available network resources. Conflicting user requirements, competition for computation and energy resources among users, and, interference in communication channels are some of the contributing factors, which dictate POMH resource allocation decisions.

Recently, matching theory has been in the spotlight to model and solve a large class of tasks offloading problems. It is a powerful mathematical tool with low complexity that can make offloading decisions based on strict preference orders for players of the opposite side. It produces stable matching assignments, in which every player is content and has no motivation to get a different allocation. Matching theory with its attractive features has not been able to gain much attention from POMH researchers.

We could only find the work of Zu et al. in [11], which uses matching theory for POMH. However, this work did not incorporate externalities in the matching process, without which, we believe that matching theory cannot be used for POMH-based allocation of computational resources.

In POMH, each subtask is processed at a different location. The sub-task that takes the longest time to compute defines the overall task delay. For efficient utilization of allocated resources, subtasks computation must be finished concurrently. This necessitates adjustment of sub-task sizes based on multiple inter-dependent factors like (a) task size, (b) computation capability of offloading and offloaded devices, (c) number of helper devices that have committed their resources to compute the task, etc. This implies that during the matching process, subtask sizes are changed at each step of the allocation. As a result, related time delay calculations of helper devices for offloading device tasks change the preference profile of helper devices. This is a case of externalities in matching theory, where preference profiles of helper devices keep on changing. As a result, matching at one helper device affects the matching of all other helper devices.

In this paper, we have developed a novel many-to-many Externalities-based Matching Algorithm (EMA), which is specifically designed to solve externalities problems associated with POMH-based task offloading. EMA always makes stable matching assignments.

To the best of our knowledge, we are the first ones to use the many-to-many matching technique with externalities to solve the resource allocation problem in the POMH scenario. The main contributions of this paper can be summarized as:

1) We formulate the resource allocation problem as a many-to-many matching problem, and ensure that subtasks complete their execution at the same time, thus enhancing the utilization of allocated resources.
2) We propose a novel many-to-many matching algorithm, specially designed to address externalities associated with the POMH problem.
3) Our literature review suggests that we are the first ones to use the many-to-many matching technique with externalities to solve the POMH problem.

The paper organization is as follows. Section II presents a review of the current work in the literature. Section III describes the system model and Section IV explains the proposed technique. Details about results and relevant discussion is given in Section V. Conclusions are highlighted in Section VI.

II. LITERATURE REVIEW

Task offloading has been a major challenge in fog computing-based IoT networks and several works have addressed this problem.

In [11], a stable matching-based technique known as SMETO is proposed to minimize energy consumption during the task offloading process. A many-to-one matching algorithm is proposed to allocate fog node resources to the tasks. The preference list is developed based on two metrics, the first measures the service level that can be provided to the tasks, and the second is the energy consumption for computation of the task. The work proposed two algorithms, the first is based on Deferred Acceptance. The second algorithm assigns several tasks to the helper nodes based on energy ranking. Results show improved energy efficiency achieved by the proposed technique.

The work in [12] proposes an algorithm to achieve energy efficiency while maintaining fairness in the network. The focus of the work is on battery-operated fog nodes. The work considers the energy consumption history of fog nodes, current task energy, and priority for making offloading decisions. An optimization algorithm is used to minimize energy consumption for task offloading. Results indicate that the proposed technique reduces the energy consumption of the fog nodes and achieves fair task offloading.

In [13], a fog task offloading algorithm known as POST is proposed. The work considers splitting tasks into subtasks and converts this problem into a Nash equilibrium problem. Gauss-Seidel algorithm is used to find out task offloading decision that meets Nash equilibrium criteria. The work shows improvement in terms of task delay.

In [14], a many-to-one matching algorithm is used for offloading tasks on different fog nodes. It is considered that the CPU of each fog node has Virtual Resource Units (VRUs) to compute the tasks. To meet task deadlines and reduce task outages, a stable matching technique combined with variable VRU sizing is proposed. Results show significant improvement in terms of reducing task outages.

The work in [15] proposes a parallel offloading technique in which fog nodes are considered to accept only a single task. A many-to-one matching technique is used to allocate subtasks to the fog nodes. To handle externalities, the JM algorithm is used to remove unstable matching pairs. Simulation results show improved computational delay at different network densities.

As compared to the previous techniques in the literature, there are two unique aspects of our proposed technique. We consider that a fog node accepts several computation tasks based on its computational capacity and utilizes a many-to-many matching technique for subtasks to fog node VRU allocation. Furthermore, we propose a novel technique that solves the externalities problem for many-to-many-matching-based POMH.

III. SYSTEM MODEL

The system model in this paper considers a fog-enabled IoT network consisting of two types of fog nodes as shown in Fig. 1. The first node is Task Nodes (TNs) that have tasks that require computation. The second node type is the Helper Nodes (HNs) which have better processing and storage capabilities and provide services to the TNs. The role of HNs and TNs is pre-defined and can not be changed. The HNs are sited to provide maximum assistance to TNs in completing their tasks. There also exists a central controller termed a Fog Node Controller (FNC) that coordinates the assignment of TN tasks to HNs. We consider that the number of TNs is m and the number of HNs is k. In this study, the POMH offloading situation has been taken into account. We assume that all tasks produced by TNs are generic split-able tasks, which may be broken down into any number of heterogeneous-sized sub-tasks. Face detection and image processing are examples of such split-able tasks [13]. (Note: The method of task division is outside the purview of our work.)

We define the size of each original task generated by a TN as W_m. With POMH, the task is split into $r+1$ smaller subtasks. The size of each subtask is represented by S_m. One of the tasks is computed locally at the TN. In percentage terms, the local computed task is α_{loc} percent of the total task size. Similarly, α_k represents the task percentage of offloaded tasks at the k^{th} HN. Thus, the sum of the local and the offloaded components add up to make the complete task.

$$\alpha_{loc} + \sum_{k=1}^{r} \alpha_k = 1 \tag{1}$$

It is assumed that the storage capacity of the HNs is divided into Virtual Resource Units (VRUs). We represent free VRUs at a HN by q_k. A HN maintains homogeneity in its VRU sizes and to account for variation, different HNs demonstrate heterogeneity in their VRU sizes and numbers.

A. COMMUNICATION MODEL

A single wireless link is required to offload the sub-tasks from one HN to a single TN. When resources of r number of HNs are allocated to a TN, r number of wireless links are required to offload sub-tasks to respective HNs. Arranging for such several wireless links is a difficult proposition to handle. Therefore, to allow contention-free access to wireless links, we resort to Orthogonal Frequency Division Multiple Access (OFDMA) technique, in which fixed channel bandwidth is allocated in an average way to all the users, i.e., if HN_k has bandwidth B_k and is assigned q_k number of sub-tasks, then each assigned TN task will get B_k/q_k amount of bandwidth resources [16], [17].

The transmission delay for a single subtask is given as;

$$T_k = \frac{\alpha_k W_m}{R_k} \tag{2}$$

Here the data rate for m^{th} TN and k^{th} HN is represented by R_k. In OFDM, R_k can be calculated as:

$$R_k = \frac{B_k}{q_k} log_2 \left(1 + \frac{P_m^t g_k}{\sigma^2}\right) \tag{3}$$

Here P_m^t is the m^{th} transceiver's transmission power, the channel gain that considers path loss and signal attenuation is represented by g_k, and noise power is represented by σ [18].

OFDMA enables HN to simultaneously receive sub-tasks from multiple TNs. However, for TN, we assume that there is a single antenna that can only transmit the sub-tasks serially. Therefore, sub-tasks have to wait for their turn for transmission, let us call this time as waiting time. The waiting time for a subtask can be given as:

$$T_{wait} = W_m \sum_{k=1}^{k-1} \frac{\alpha_k}{R_k} \tag{4}$$

The time taken to transmit all subtasks can be calculated using Eq. 2 and Eq. 4 as:

$$T_m^{tx} = W_m \sum_{k=1}^{r-1} \frac{\alpha_k}{R_k} + \frac{\alpha_k W_m}{R_k} \tag{5}$$

B. TIME DELAY MODEL

The time delay of the task is divided into two main parts, the local computation delay and the HN computation delay.

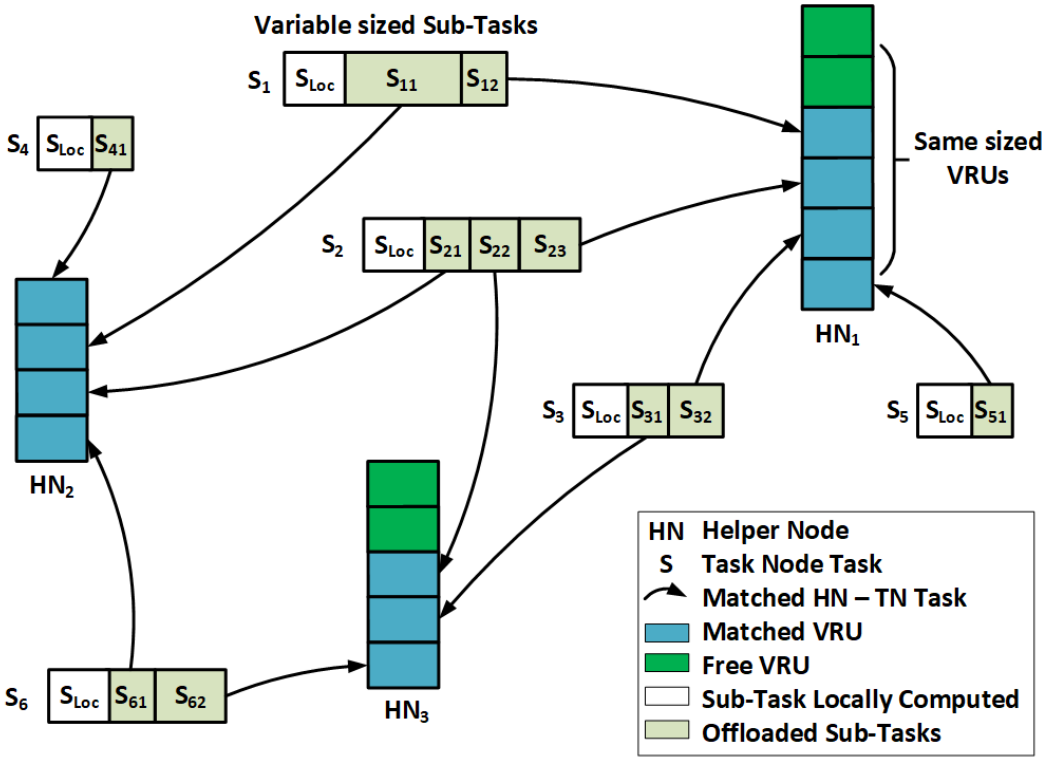


FIGURE 1. System model.

1) LOCAL COMPUTATION DELAY

The local computation delay T_m^l at a TN can be given as:

$$T_m^l = \frac{\alpha_{loc} W_m C_r}{C_m} \tag{6}$$

Here C_r is the number of cycles in which the task can be computed and C_m is the speed of TN's CPU.

2) HN COMPUTATION DELAY

The offloaded computation delay T_k^c at HNs can be given as:

$$T_k^c = \frac{\alpha_k W_m C_r}{C_k} \tag{7}$$

Here C_r is the number of cycles in which the task can be computed and C_k is the speed of HN's CPU.

In the time delay model, we ignore any queuing delay. We also consider that the time taken to transmit the result of the task is negligible [19].

The computation delay for the offloaded task by HN can be given as (1) when it is the sole resource donor to task W_m and, (2) when other HNs are also contributing towards computing W_m is given by Eq. 8 and Eq. 9, respectively:

$$T_k^m = \frac{\alpha_k W_m}{R_k} + \frac{\alpha_k W_m C_r}{C_k} \tag{8}$$

$$T_k^m = \frac{\alpha_k W_m}{R_k} + \frac{\alpha_k W_m C_r}{C_k} + W_m \sum_{k=1}^{k-1} \frac{\alpha_k}{R_k} \tag{9}$$

3) TOTAL DELAY

Every sub-task will experience different delays. The total task delay is dependent on the sub-task that is computed at the end.

$$T_m = maximum \left\{ T_m^l, T_k^m \right\} \tag{10}$$

For the task to be completed, the task delay must be less than the task deadline. i.e., $T_m \leq T_m^{max}$

C. SIZING α's TO COMPLETE SUB-TASKS AT SAME TIME

Careful analysis of Eq. 10 reveals that we can achieve effective utilization of allocated resources if the time delay of all subtasks is the same. To achieve this, sub-task sizes need to be adjusted such that the following equation is satisfied.

$$T_m^l = T_1^m = T_2^m \ldots = T_r^m \tag{11}$$

In the case of single HN, the value of α_k at which equal time delay is achieved can be found by equating Eq. 6 to Eq. 8 as:

$$\alpha_k = \frac{\alpha_{loc} R_k C_r C_k}{C_m (C_k + R_k C_r)} \tag{12}$$

The corresponding value of α_{loc} can be found by using value of α_k from Eq. 12 in Eq. 1, as:

$$\alpha_{loc} = 1 / \left(1 + \frac{R_k C_r C_k}{C_m (C_k + R_k C_r)} \right) \tag{13}$$

Similarly, for r number of HNs, $r+1$ number of equations can be derived. By solving these equations, we can find the

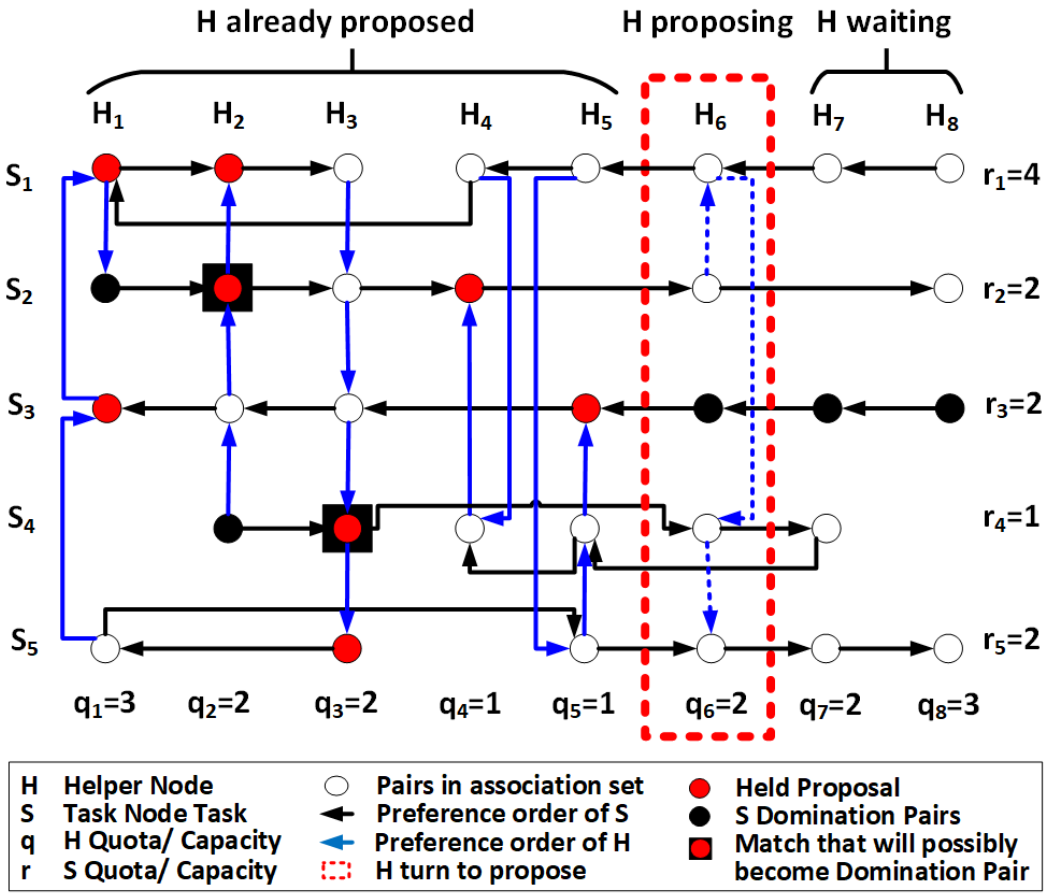


FIGURE 2. Proposed many-to-many matching algorithm.

inter-dependent values of all α's such that, they all complete at the same time.

IV. PROPOSED TECHNIQUE

In this section, we explain in detail the working of the proposed technique.

A. SUMMARY OF THE TECHNIQUE

The goal of the proposed technique is to minimize the time delay of the task offloading by using partial offloading to multiple helpers in the IoT network. To achieve this task, we utilize many-to-many matching to allocate subtasks of TNs to the VRUs of the HNs. The developed matching scenario is faced with externalities problems caused by to dynamically changing preference profile of HNs. To solve this problem, we propose a novel algorithm that handles the dynamic preference profiling issue of HNs on the fly while matching TN sub-tasks to HN resources. We call our proposed algorithm an Externalities-based Matching Algorithm (EMA).

The working process of the proposed EMA is shown in Fig. 2. TN tasks are shown as row players HNs are shown as column players and their preference profile is shown as horizontal and vertical arcs, respectively.

B. PREFERENCE PROFILES

In the proposed EMA, TN tasks have strict preference profile $\succ_S$ over the HNs whereas no pre-hand preference profile is set for HNs. The preference of TN towards HN is based on task computation time, thus the HN which provides the lowest computation time is ranked the highest in preference by the TN.

On the other hand, the HN node wants to minimize the network-level task computation time. For this purpose, the HN calculates the difference in computation time when the task is allocated to the HN and when the task is allocated to some other HN. Thus, HN prefers TNs for which computation time advantage is highest. For POMH, all subtasks are required to be completed at the same time (from equation 11). During the matching process, the offloaded task percentages α_k change at each iteration, and the preference of HNs changes dynamically leading to externalities.

C. SOLVING EXTERNALITIES AND AVOIDING CYCLES

Unlike work in [20] and [21], where the externalities problem is solved after the initial stable matching assignments are obtained, EMA solves the externalities problem during the matching process. For this HNs on their turn to propose,

Algorithm 1 Proposed EMA

Input: C_k, C_r, C_m, T_m^{max}, H_k, $\succ_{S_m}$, S_m
Output: Matching allocation: λ_{out}
$\lambda_{in} = [\,]$
while $(\lambda_{in} \neq \lambda_{out})$ **do**
  for $\forall H$ **do**
    for $\forall (S \in H_k)$ **do**
      if ($H_k \notin$ *blocking pair of* S_m) **then**
        if ($|$ *held matches* $|== r_m$) **then**
          Note held match, lowest in $\succ_{S_m}$
          Find % improvement in task time
        else
          Find % improvement in task time
        end
      else
        Do not propose S_m
      end
    end
    Propose q_k tasks with best % time improvement
    Define noted match (if any) as blocking pair
    if (H_k *does not propose* $S_{m'}$) **then**
      Delete H_k from $S_{m'}$ held proposals
      Reset blocking pairs of $S_{m'}$
    end
  end
  $\lambda_{in} = \lambda_{out}$
end

calculate the priority order of TN tasks based on the percentage time improvement the task will experience, if accepted by HN.

Before proposing, HN also checks the blocking pair (if two nodes prefer each other instead of their current matching allocation) list of TN tasks and proposes TN tasks where their request is guaranteed to be accepted. As a result, algorithm convergence time is reduced, as are the chances of developing cyclic patterns (the scenario where a single blocking pair generates another one and the process continues and enters a loop), as expected by Knuth [22].

D. WORKING OF PROPOSED EMA

Proposed EMA is based on polyandrous polygamy algorithm [23]. We have modified this algorithm to solve the externalities problem by deferring proposal acceptance until finished, i.e., we have incorporated features of the Deferred Acceptance Algorithm (DAA) in EMA [24].

In the proposed EMA, HNs have dynamic preference profiles and are the only ones allowed to make proposals. To address the externalities problem, HNs work out their preference profile $\succ_H$ at the time of proposal making and make calculated matching decisions. HN H_k can propose q_k number of TN tasks, whereas, TN task S_m can hold up to r_m number of proposals.

When the number of held proposals equals r_m, S_m examines its preference profile $\succ_{S_m}$ and classifies all HNs with a priority lower than the lowest priority held proposal, as dominating/ blocking pairs. A TN task will not be matched to its blocking pairs.

Due to the dynamic nature of HN preference profiles, there is an equal probability that an HN H_k may not propose the same TN task S_m in the next iteration. In this situation, the number of proposals held by S_m, against which it had defined blocking pairs, will become less than its allowed quota r_m. If this happens, all HNs defined as their blocking pairs will be unblocked and S_m will be able to match all HNs.

When the number of held proposals equals r_m again, the TN task will start redefining its blocking pairs. Therefore, in the proposed EMA, the TN tasks blocking pairs are not permanent. The HNs continue to be defined within and outside of the blocking pairs.

E. STEP BY STEP PROCEDURE OF EMA

The proposed EMA takes the following actions to find stable matching assignments in polynomial time:

- Initial matching assignments are set to empty set.
- On its turn to make proposals, HN H_k finds the ranking of TN tasks in terms of preference. For this, H_k first examines the blocking pair list of TN task S_m to see if H_k is defined in that list or not. If H_k is defined as a blocking pair, it does not propose S_m.
- If H_k is not defined in the blocking pair list of S_m, H_k checks the number of proposals held by S_m and does following:
  - If the number of held proposals with S_m equals its quota r_m, the lowest preferred held proposal from $\succ_{S_m}$ will be deleted and defined as blocking pair, if H_k proposes S_m. This held proposal is marked individually. Leaving beside this proposal, H_k calculates α_k and percentage improvement in task completion time, if S_m is computed by H_k.
  - If the number of held proposals with S_m is less then its quota r_m, H_k considers all held proposals with S_m and calculates α_k and percentage improvement in task completion time, if S_m is computed by H_k.
- H_k shortlists q_k number of TN tasks with the best percentage improvement in time and proposes them.
- If proposed TN task S_m had marked held proposal, that held proposal is rejected and defined as a blocking pair.
- If H_k does not proposes a TN task $S_{m'}$, which it had proposed last time, H_k match with $S_{m'}$ is deleted. All blocking pairs of $S_{m'}$ are reset.
- Initial matching assignments are set equal to the assignment that is matched at the end of the iteration.

The above algorithm repeats till the time a balance is achieved, i.e., the initial matching assignment is equal to the assignment that is matched at the end of the iteration.

TABLE 1. Simulation settings.

Description	Value
Value of TNs	10-120
Value of HNs	10
Computational capacity of HNs	4-5 GHz
Number of VRUs	Number of TNs x r
Computational capacity of TNs	0.8-1.2 GHz
Size of the tasks	4000-5000 KB
Task computational requirement	2500-3000 Hz
TN transceiver's power	100 mW
Bandwidth	5 MHz

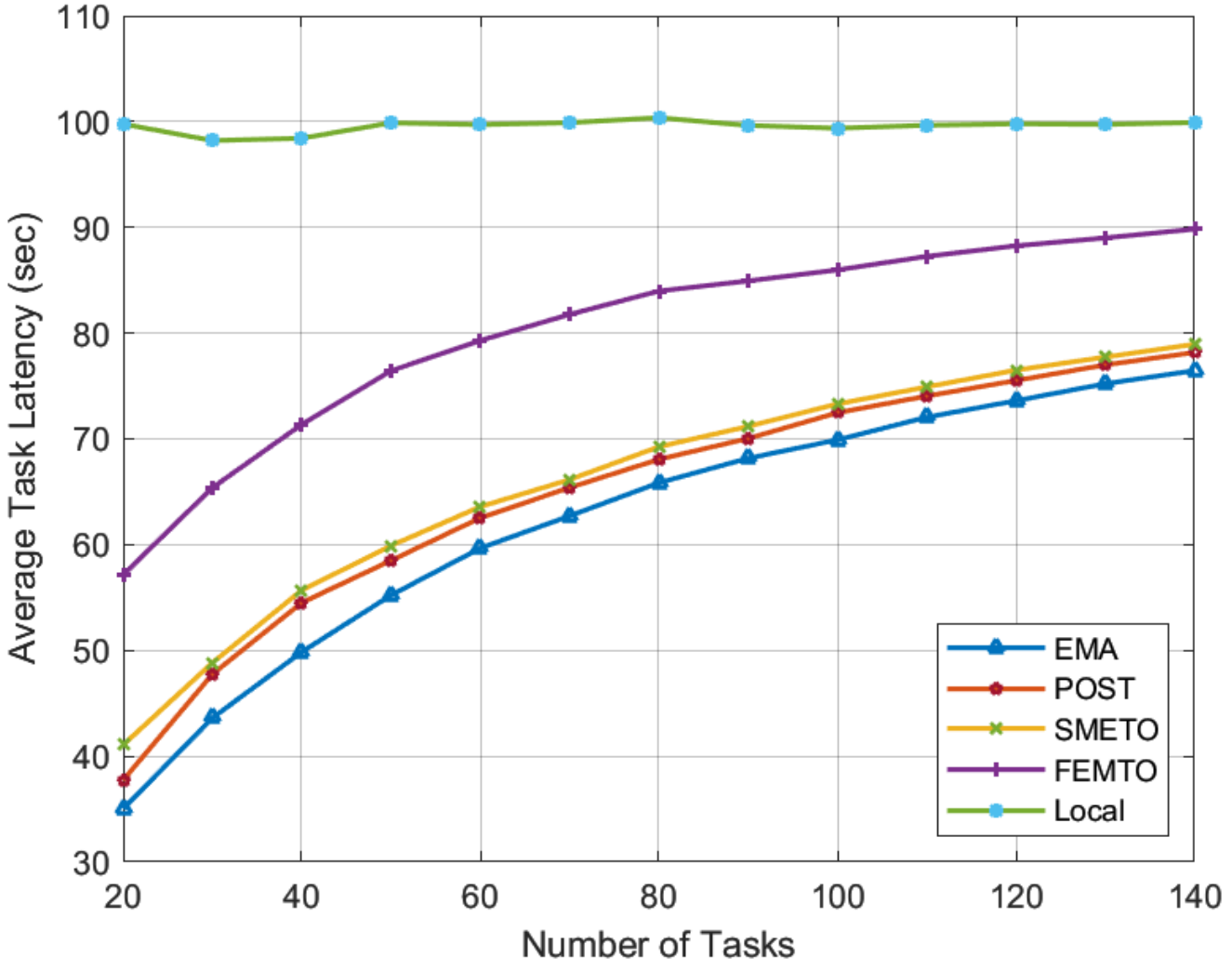


FIGURE 3. Average Task Latency with Four Task Splits.

V. RESULTS AND DISCUSSION

We present a detailed simulation-based evaluation of the proposed EMA technique in this section.

A. SIMULATION SCENARIO

A simulation setup was developed that consists of a fog-based IoT network using MATLAB. The key values of the

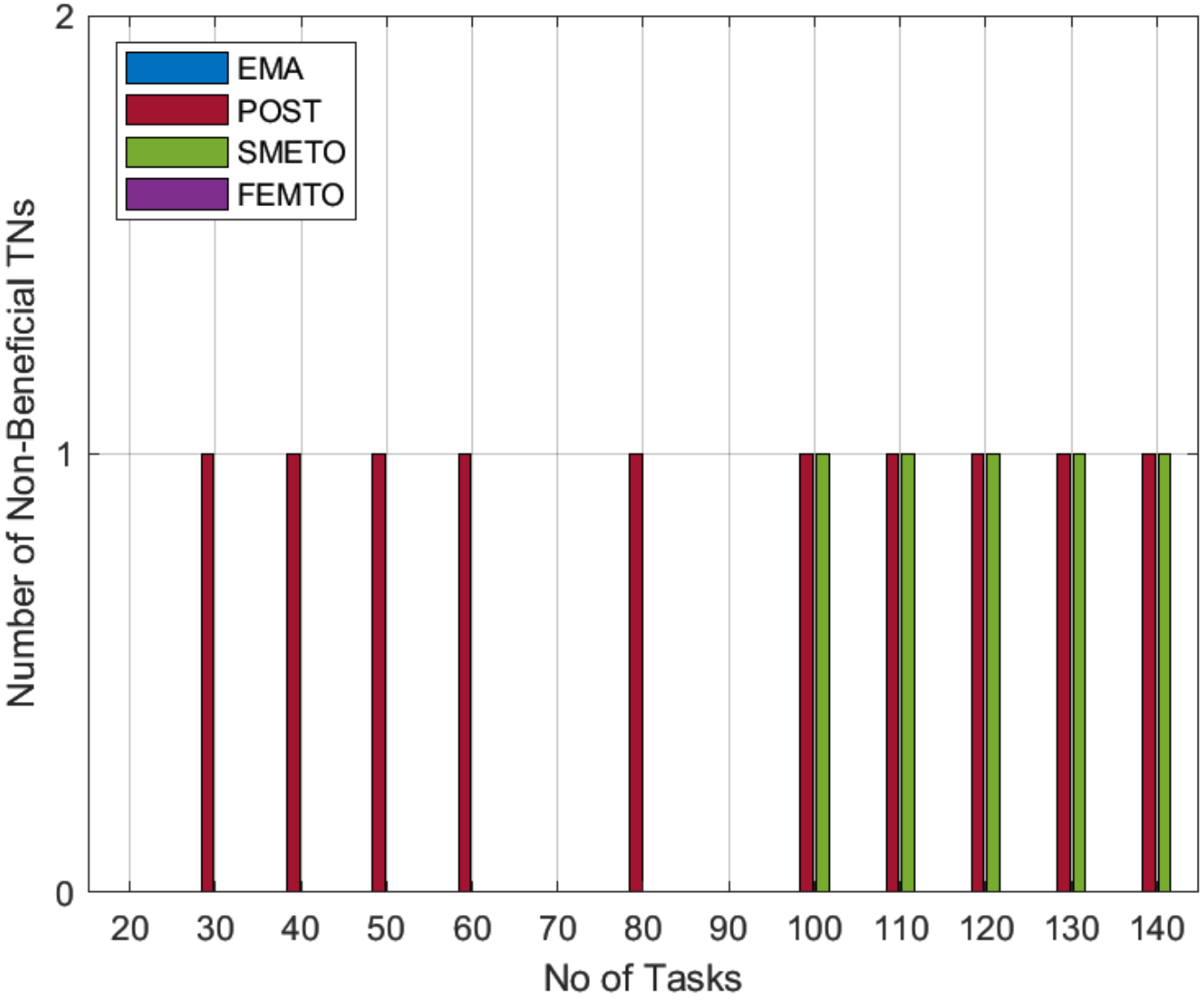


FIGURE 4. Number of TNs which are non-beneficial as a result of offloading.

simulation parameters are listed in Table 1. The network consists of 10 HNs, which are evenly distributed within a geographical area of 60 m x 60 m. The computational capacity of HNs varies for every node and has a minimum value of 4 GHz and a maximum value of 5 GHz. To test the performance of EMA at different network densities, the TNs are varied between a minimum value of 10 and a maximum value of 120. These TNS are considered to be generally scattered in the area. The computational capacity of TNs is set to be between 0.8 GHz to 1.2 GHz.

For task splitting, we consider the maximum possible subtasks to be 6. However, the exact number depends on the number of TN to HN matches obtained. Out of these subtasks, one task is computed locally whereas the others are processed at the HNs. The subtask size is selected such that the computation time of each of them is the same. To accommodate for many-to-many matching and utilize complete resources of HNs, the VRUs numbering is selected to be equal to the total number of sub-tasks, the TNs need to offload. The bandwidth for task transmission from TN to HN is 5 MB. The TN transceiver's transmission power is selected as 100 mW. The free space path loss model is used as per the reference [25].

B. COMPARISON OF EMA WITH OTHER TECHNIQUES

The performance of the proposed EMA is compared with four other techniques namely, SMETO [11], FEMTO [12], POST [13] and Local (in which all tasks are computed by TNs themselves). The working of SMETO, FEMTO, and POST is described already in Section II. Both SMETO and FEMTO aim to achieve energy efficiency, whereas, POST aims to achieve time efficiency. For simulation purposes, we have considered parallel computation of tasks in POST. Resource allocation for POST and FEMTO is replicated

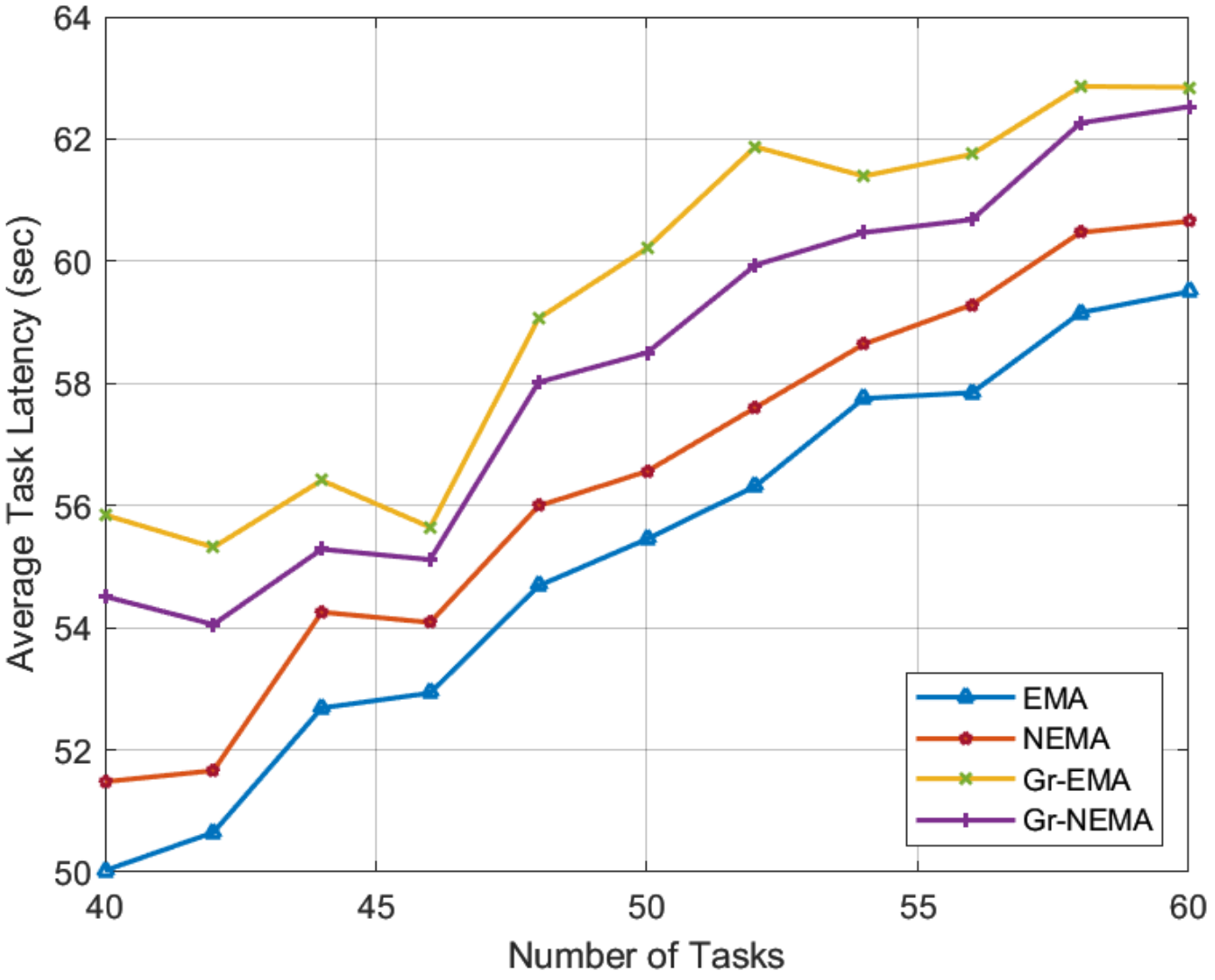


FIGURE 5. Average Task Latency - Varying HN Objective Function.

through Polyandrous Polygamy (many-to-many) and DAA (many-to-one) matching techniques, respectively.

The evaluation parameters used is this paper to evaluate the proposed scheme and other baseline schemes are task latency and number of non-beneficial tasks. Task latency measures the time taken to complete each task, offering insights into the system's efficiency and how well the objective functions are met by the preference profile parameters. Non-beneficial tasks are those that fail to find a match and must be performed by the device generating the task. This evaluation criterion also provides valuable insights into the fairness of the schemes, based on their ability to minimize the number of unserved tasks.

In Fig. 3, the plot of average latency or delay of tasks is presented. It can be seen that the proposed EMA performs better in terms of efficiency as compared to the other schemes. The results of the proposed EMA are better than other baseline schemes due to: (1) We have defined the HN objective function to achieve network-level time efficiency. As a result, HNs prefer tasks that contribute more towards network time efficiency. This objective function generates the best time efficiency results, with or, without solving externalities problems, (2) In the EMA algorithm, HNs re-calculate time efficiency before making every proposal and thus always make an informed matching decision and improve overall time efficiency results.

In comparison, POST tries to improve task delay reduction for both TNs and HNs and, SMETO aims to reduce energy consumption in the network. The difference in results of the proposed EMA and these baseline schemes is primarily attributed due to the difference in the objective function of HNs. Otherwise, all these schemes re-adjust sub-task sizes, such that they all complete at the same time. FRETO aims for energy efficiency and also ensures equal time completion of all sub-tasks. The advantage of FRETO over other techniques

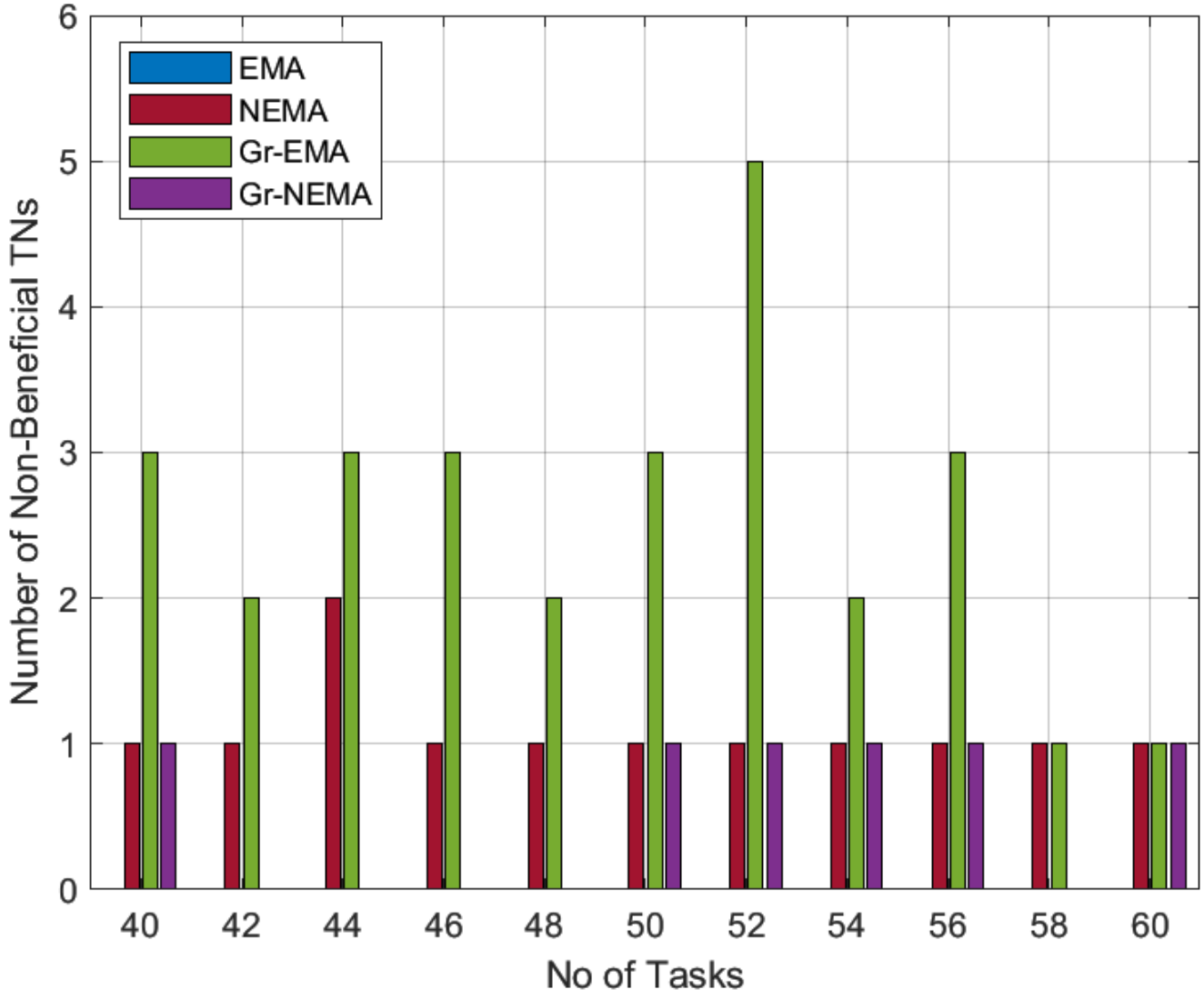


FIGURE 6. Number of TNs which are non-beneficial as a result of offloading.

is the partial offloading technique in which the task is offloaded to one HN only. Thus, FRETO has comparatively lower results than other baseline schemes.

Fig. 4 plots the TNs for which the offloading technique did not reduce the task delay as compared to local computation. As discussed earlier, HNs in preference profiling for network time efficiency favor lonely tasks. With the results, it is confirmed that by using preferences based on network time efficiency, most TNs get served.

C. RESULTS RELATED TO PERFORMANCE OF EMA IN THE PRESENCE OF EXTERNALITIES

To analyze the performance of EMA in the presence of externalities, a comparison is done with: (1) Non-externality-based many-to-many matching technique (NEMA), (2) Externalities-based many-to-many matching technique, where HNs preference profile is based on greedy approach (Gr-EMA), (3) Non-externality based many-to-many matching technique, where HNs preference profile is based on greedy approach (Gr-NEMA),

1) HNS PREFERENCE PROFILING AND TASK LATENCY

The externalities problem exists in many-to-many based POMH offloading as it is necessary to ensure that subtasks are computed together time-wise. Therefore, when the amount of allocated resources changes with the allocation and cancellation of matches during the matching process, the sub-task sizing requires adaptation to achieve the same finish time. As a result, time delay and power calculations of HNs for TN tasks change resulting in a change in HNs preference profile. This dynamically changing preference profile of HNs creates the externalities problem. It is important to highlight here, that the reason for solving the externalities problem is to achieve stability in matching decisions by satisfying objective functions of all players. The solution of the externalities problem will enhance time or energy

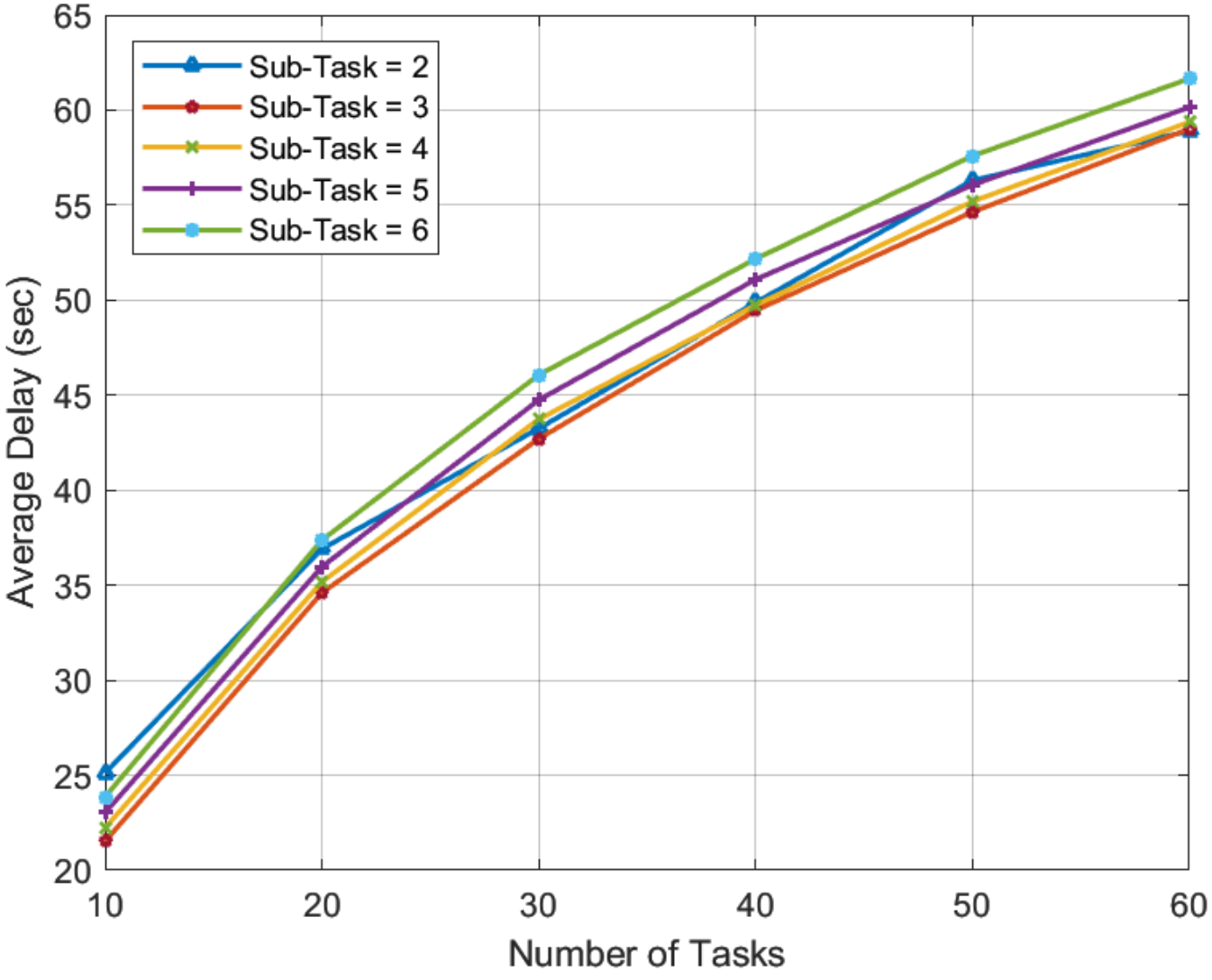


FIGURE 7. Task Efficiency with Different Number of Task Splits.

efficiency, only, when the objective functions of players are in sync with the overall objective of the resource allocation problem. To highlight this point, we have considered two HN preference profiling techniques that produce opposite results when the externalities problem is solved. These techniques are discussed below:

Firstly, we consider the preference method in which HN preference is set in ascending order concerning the time delay required for task computation. This technique can be termed a greedy preference profiling technique because HNs are looking to optimize their time only. When we solve this externalities problem, a matching scenario is generated in which HNs favor either tasks with small size or tasks that win higher matches. With this matching trend, HNs end up contributing less towards TN tasks. Therefore, this objective function generates poor time efficiency results against the non-externalities-based matching technique, as evident from results in Fig. 5.

Whereas in this paper, we have set HN's objective function to achieve network-level time efficiency. By doing so, the objective function of all players aligns with the overall objective function of the resource allocation problem.

In this preference profiling technique, HNs favor lonely tasks and those tasks, serving which contribute more towards network time efficiency. Now the externalities-based solution gives the best time efficiency results. It is important to highlight here, that solving externalities problems generally improves task efficiency. However, Ma [26] proposed that solving for externalities problem gives a variety of stable matching in each iteration. Because of this variety, there is a remote possibility that task efficiency may not improve in a few odd cases when compared with non-externality-based matching results.

The effect of preference profiling techniques on non-stable matching is also plotted in Fig. 5. The results show that the preference profiling for network time efficiency outperforms

the greedy preference profiling technique. We can safely conclude that the greedy preference profiling technique benefits some HNs but increases overall task completion time. Whereas, preference profiling for network time efficiency improves the time efficiency of all TNs and HNs.

2) HN PREFERENCE PROFILING AND NUMBER OF NON-BENEFICIAL TNs

The results in Fig. 6, reflect the matching trends generated by the preference techniques. As discussed earlier, HNs in preference profiling for network time efficiency favor lonely tasks. It can be seen that using network time efficiency-based preference causes the most number of TNs to get served in both externalities and non-externalities-based techniques. Whereas in greedy preference profiling, HNs favor tasks with more number matches and shy away from lonely tasks. The results confer the matching trend and, we find that the greedy preference profiling technique comparatively serves less number of TNs.

3) NUMBER OF TASK SPLITS

The purpose of Fig. 7 is to understand the limits of the POMH task offloading process and to show how different numbers of subtasks impact the average delay. The results show that when the number of task splits is increased to exploit the advantage of parallel task computation, the time efficiency does not improve proportionately. Rather, there is a limit to which a task can be beneficially decomposed into sub-tasks to achieve time efficiency. The primary underlying reason for this deterioration in time efficiency is the fact, that when a task is split into more sub-tasks (let's say 5), a TN needs to transmit 4 sub-tasks on 4 different wireless channels to their respective HNs. These HNs will have different distances and different channel rates with the offloading TN. Therefore, the offloaded sub-tasks will have different transmission times. The resources provided by multiple HNs will be able to overcome this increase in transmission time to a certain limit, after which, POMH starts giving negative results, i.e., takes more time to complete the tasks. Thus, there is a need to develop an algorithm, that can adaptively manage the number of task splits to provide the best time efficiency results.

VI. CONCLUSION

This paper presents a novel technique for partial offloading of tasks to multiple helper nodes in fog-enabled IoT networks. The helper node computational resource allocation problem is solved using a many-to-many matching technique. The preference profiles of both task-generating nodes and helper nodes are based on improving the task computational delay. Furthermore, the proposed technique suffers from externalities problems due to the dynamic preference profile of helper nodes. The proposed technique solves this issue by developing an adaptive matching scheme that removes blocking pairs within the matching process. Results show up to 70% improvement in terms of task latency by the proposed scheme. In the future, we will focus on further extension of this technique in scenarios where helper nodes can suffer from faults

REFERENCES

[1] 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.

[2] M. Goudarzi, M. Palaniswami, and R. Buyya, "A distributed deep reinforcement learning technique for application placement in edge and fog computing environments," *IEEE Trans. Mobile Comput.*, vol. 22, no. 5, pp. 2491–2505, May 2023.

[3] Z. Lin, L. Lu, J. Shuai, H. Zhao, and A. Shahidinejad, "An efficient and autonomous planning scheme for deploying IoT services in fog computing: A metaheuristic-based approach," *IEEE Trans. Computat. Social Syst.*, vol. 11, no. 1, pp. 1415–1429, Feb. 2024.

[4] M. D. Hossain, T. Sultana, M. A. Hossain, M. I. Hossain, L. N. T. Huynh, J. Park, and E.-N. Huh, "Fuzzy decision-based efficient task offloading management scheme in multi-tier MEC-enabled networks," *Sensors*, vol. 21, no. 4, p. 1484, Feb. 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/4/1484

[5] L. Sun, G. Xue, and R. Yu, "TAFS: A truthful auction for IoT application offloading in fog computing networks," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 3252–3263, Feb. 2023.

[6] S. Mousavi, S. E. Mood, A. Souri, and M. M. Javidi, "Directed search: A new operator in NSGA-II for task scheduling in IoT based on cloud-fog computing," *IEEE Trans. Cloud Comput.*, vol. 11, no. 2, pp. 2144–2157, Apr./Jun. 2023.

[7] 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.

[8] L. N. T. Huynh, Q.-V. Pham, X.-Q. Pham, T. D. T. Nguyen, M. D. Hossain, and E.-N. Huh, "Efficient computation offloading in multi-tier multi-access edge computing systems: A particle swarm optimization approach," *Appl. Sci.*, vol. 10, no. 1, p. 203, Dec. 2019. [Online]. Available: https://www.mdpi.com/2076-3417/10/1/203

[9] X. Wang, Z. Ning, M. Zhou, X. Hu, L. Wang, B. Hu, R. Y. K. Kwok, and Y. Guo, "A privacy-preserving message forwarding framework for opportunistic cloud of things," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 5281–5295, Dec. 2018.

[10] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 3rd Quart., 2017.

[11] Y. Zu, F. Shen, F. Yan, L. Shen, F. Qin, and R. Yang, "SMETO: Stable matching for energy-minimized task offloading in cloud-fog networks," in *Proc. IEEE 90th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2019, pp. 1–5.

[12] G. Zhang, F. Shen, Z. Liu, Y. Yang, K. Wang, and M.-T. Zhou, "FEMTO: Fair and energy-minimized task offloading for fog-enabled IoT networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4388–4400, Jun. 2019.

[13] Z. Liu, Y. Yang, K. Wang, Z. Shao, and J. Zhang, "POST: Parallel offloading of splittable tasks in heterogeneous fog networks," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3170–3183, Apr. 2020.

[14] U. M. Malik and M. A. Javed, "Ambient intelligence assisted fog computing for industrial IoT applications," *Comput. Commun.*, vol. 196, pp. 117–128, Dec. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366422003735

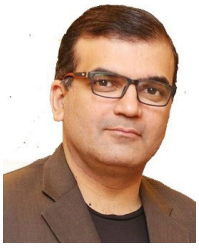
[15] U. M. Malik, M. A. Javed, J. Frnda, J. Rozhon, and W. U. Khan, "Efficient matching-based parallel task offloading in IoT networks," *Sensors*, vol. 22, no. 18, p. 6906, Sep. 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/18/6906

[16] 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.

[17] J. Wang, T. Lv, P. Huang, and P. T. Mathiopoulos, "Mobility-aware partial computation offloading in vehicular networks: A deep reinforcement learning based scheme," *China Commun.*, vol. 17, no. 10, pp. 31–49, Oct. 2020.

[18] 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.

[19] X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser joint task offloading and resource optimization in proximate clouds," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3435–3447, Apr. 2017.
[20] A. Alimudin and Y. Ishida, "Matching-updating mechanism: A solution for the stable marriage problem with dynamic preferences," *Entropy*, vol. 24, no. 2, p. 263, Feb. 2022.
[21] V. Kanade, N. Leonardos, and F. Magniez, "Stable matching with evolving preferences," 2016, *arXiv:1509.01988*.
[22] D. E. Knuth, *Stable Marriage and Its Relation to Other Combinatorial Problems: An Introduction to the Mathematical Analysis of Algorithms*. Providence, RI, USA: American Mathematical Society, 1997.
[23] M. Baiou and M. Balinski, "Many-to-many matching: Stable polyandrous polygamy (or polygamous polyandry)," *Discrete Appl. Math.*, vol. 101, nos. 1–3, pp. 1–12, Apr. 2000.
[24] A. E. Roth, "Deferred acceptance algorithms: History, theory, practice, and open questions," Dept. Econ., Nat. Bur. Econ. Res. (NBER), Stanford Univ., Stanford, CA, USA, Work. Paper 13225, Jul. 2007.
[25] 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.
[26] J. Ma, "On randomized matching mechanisms," *Econ. Theory*, vol. 8, no. 2, pp. 377–381, Aug. 1996.



USMAN MAHMOOD MALIK received the B.S. degree in electrical (telecom) engineering, in 2000, the M.S. degree in system engineering from the National University of Science and Technology (NUST), Pakistan, in 2016, and the Ph.D. degree in electrical engineering from COMSATS University Islamabad, Pakistan, in 2023. He is currently working as a Faculty Member with the Department of Computer Science Engineering, NUST. His research interests include modeling and system designing, vehicular and the 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 worked as a Postdoctoral Research Scientist at 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.

ABDULAZIZ ALMOHIMEED received the master's degree from Monash University, Australia, and the Ph.D. degree from the University of Southampton, U.K. He is currently working as an Assistant Professor with the College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia. His research interests include natural language processing, artificial intelligence, data science, the Internet of Things, and network security. He is passionate about leveraging technology to create innovative solutions.

MOHAMMED ALKHATHAMI is currently working as an Associate Professor with the Department of Information Systems, College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia. His research interests include communication systems, networks, security, and computing.

ABEER ALMUJALLI received the bachelor's and master's degrees from King Saud University, Saudi Arabia. She is currently working as a Lecturer with the College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia. Her research interests include networks, communications, and security.