Version of Record: https://www.sciencedirect.com/science/article/pii/S157411922030122X
Manuscript_d70681e26ac84039588fe9fb99dd1039

Bandwidth-Constrained Task Throughput Maximization in IoT-enabled 5G Networks

Ajay Pratap[†], Ragini Gupta[‡], Venkata Sriram Siddhardh Nadendla[‡], Sajal K. Das[‡]

[†] *Department of Computer Science and Engineering, Indian Institute of Technology (BHU) Varanasi, India*
[‡] *Department of Computer Science, Missouri University of Science and Technology, Rolla, MO, USA*

Abstract

Fog computing in 5G networks has played a significant role in increasing the number of users in a given network. However, Internet-of-Things (IoT) has driven system designers towards designing heterogeneous networks to support diverse task demands (e.g. heterogeneous tasks with different priority values) under interference constraints in the presence of limited communication and computational resources. In this paper, our goal is to maximize the total number of tasks served by an IoT-enabled 5G network, labeled *task throughput*, in the presence of heterogeneous task demands and limited resources. Since our original problem is intractable, we propose an efficient two-stage solution based on multi-graph-coloring. We analyze the computational complexity of our proposed algorithm, and prove the correctness of our algorithm. Lastly, simulation results are presented to demonstrate the effectiveness of the proposed algorithm, in comparison with state-of-the-art approaches in the literature.

Keywords: Resource Allocation, PRB, 5G, IoT, Fog, Graph.

1. Introduction

The proliferation of heterogeneous computing devices in everyday life has enabled system designers to implement Internet-of-Things (IoT) and improve services in diverse domains such as healthcare, manufacturing and transportation. Despite the availability of high-performance computing abilities, the main challenge in designing an IoT network is to provide services seamlessly in the presence of ever-increasing number of edge devices [1, 2, 3]. Therefore, fog computing has been proposed in 5G networks to reduce the computational load centrally, *i.e.*, at the base station (BS), and support the ever-increasing number of mobile IoT devices [4]. The IoT devices in fog networks generate tasks via sensing information from a physical phenomenon, and send pre-processed data to a gateway node called *fog access point* (FAP). Upon receiving the data from IoT devices, FAP executes the task and sends the response back to respective IoT devices.

Email addresses: ajay.cse@iitbhu.ac.in (Ajay Pratap[†]), g5rv@mst.edu (Ragini Gupta[‡]), nadendla@mst.edu (Venkata Sriram Siddhardh Nadendla[‡]), sdas@mst.edu (Sajal K. Das[‡])

For example, consider a smart-health application where an IoT-enabled 5G network is designed to serve stroke patients in a rehabilitation center. While it is necessary to continuously monitor various signals such as blood pressure, heart rate and blood sugar levels in multiple patients, there are other tasks such as fall detection (typically detected using accelerometers, gyroscopes and surveillance cameras) that play a crucial role in the avoidance of accidents during the rehabilitation period. Therefore, tasks such as fall detection take precedence over processing blood sugar readings. At the same time, the latency and bandwidth requirements for video streaming in surveillance cameras connected with the same 5G network are significantly larger than those needed to communicate and process fall detection data. In other words, IoT devices typically generate heterogeneous demands (multi-priority tasks) that require diverse resource requirements (e.g. bandwidth, computational power) in the presence of non-identical latency constraints. In such a 5G network, BS should optimize system efficiency via prioritizing and allocating appropriate resources (e.g. bandwidth, CPU cycles in FAPs) to different tasks dynamically, while simultaneously integrating heterogeneous constraints and dynamic network environments [5].

However, one of the main challenges in designing a BS in a 5G network is that resource reuse can potentially lead to significant interference due to high density of nodes. Such a sensitivity to interference in the presence of large number of task requests makes resource allocation, one of the most challenging tasks in the design of a 5G network. In this paper, our goal is to develop a novel, scalable and efficient resource allocation algorithm for IoT-enabled 5G networks that takes into account various issues such as task priority, interference constraints and network connectivity. Specifically, in our proposed approach, the BS identifies high priority tasks and allocates necessary resources to appropriate FAPs. If there are residual resources (or reusable resources whenever FAPs are non-interfering with each other), the BS allocates them to serve low priority tasks in the network [6].

1.1. Our Contributions

After describing the system model along with underlying constraints and interactions between IoT devices, FAPs and the BS, we formulate the problem of maximizing the total number of heterogeneous tasks (a.k.a. task throughput) served by the 5G network in the presence of heterogeneous resource demands and preference of IoT devices. Given that this is an intractable non-convex problem, we propose an efficient two-stage allocation algorithm based on multi-graph-coloring. The novelty of our solution approach lies in separating the problem into two allocations. In Stage I, the BS allocates spectrum to FAPs; and in Stage II, the FAPs distribute their allocations amongst the connected IoT devices based on the respective task priorities. We analyze the computational complexity of the proposed two-stage algorithm, the validity of its assignment and the number of wireless channels utilized. Finally, we study the performance of our algorithm and present experimental results.

The rest of the paper is organized as follows. Section 2 reviews the related works. Section 3 presents the

system model and assumptions. The problem formulation is described in Section 4 while Section 5 proposes the two-stage algorithm. Stage I of multi-coloring on interference graph is detailed in Section 6), and Stage II dealing with distribution of PRBs to IOT is presented in Section 7. Theoretical analysis is presented in Section 8, while Section 9 describes the experimental results. The final section 10 concludes the paper.

2. Related Work

Limited efforts have been made to address problems similar to the one under consideration in this paper. For example, a novel QoS-based resource management scheme for IoT devices is proposed in [7], where the authors reasoned out the need for a discrete number of resources at different stages of operation in the context of smart healthHowever, they ignored the effects of interference between densely deployed IoT devices that may cause severe interference in the presence of spectrum scarcity. On the other hand, in [8] is proposed a matching theory based pairing model for resource sharing using Irving's stable roommate algorithm. Likewise, [9], the authors addressed the resource allocation problem using a three-tier solution, which is based on Stackelberg games and matching theory. In addition to ignoring the interference constraints at various entities within the network, both these approaches rely on a completely distributed architecture which suffer from a price-of-anarchy due to deteriorated allocation efficiency.

On the contrary, several efforts [10, 11, 12] have been made recently to address resource allocation in IoT-enabled 5G networks, where IoT devices interact with FAPs and BS in order to avail communication and computational resources and complete their tasks. However, to the best of our knowledge, none of these efforts have considered resource re-usability in the solution approaches. Thus, existing approaches are not prudent in IoT-enabled fog architecture due to their convoluted computation and rigid structure of FAPs. The randomly deployed FAPs are likely to create interference to IoT due to overlapping coverage area and limited availability of resources. Unlike existing approaches, we deal with the allocation of spectrum resources at FAPs followed by distribution of resources at IoT level, while simultaneously considering the interference, non-uniform demand and priority of different tasks.

3. System Model

Consider a network shown in Fig. 1, where M heterogeneous IoT devices request bandwidth and computational resources to FAPs regarding their respective tasks. Let $\mathbb{I} = \{I_1, \cdots, I_M\}$ and $\mathbb{F} = \{F_1, \cdots, F_K\}$ denote the set of all IoT devices and FAPs in the network respectively. Assume that the i^{th} device I_i generates a single task $s_i \in \Psi$, where Ψ represents the set of tasks that the network can execute. Therefore, we use the labels *IoT device* and its corresponding *task* interchangeably henceforth in this paper. Let Φ_k denote the set of all IoT devices that are within the physical proximity of F_k. Note that Φ_k can also be interpreted as the coverage area of F_k. Therefore, it is natural for BS to assign F_k to all the IoT devices

within Φ_k in order to minimize latency. However, it is also possible that the coverage areas of two nearby FAPs can overlap, which leads to interference (consequently, a reduction in the achievable data rate) in the communication between the IoT devices within the overlap region and the corresponding FAPs [13]. Furthermore, once the BS matches an IoT device to a FAP, the IoT device shares the task details to the FAP using one or more Physical Resource Blocks (PRBs), which is the smallest unit of communication resource assigned by the BS [14]. We have considered Orthogonal Frequency Division Multiple Access (OFDMA) model where, a PRB comprises of 180 kHz bandwidth (Δf) and 0.5 ms time frame [15].

Let $\mathbb{N} = \{1, \cdots, N\}$ denote the set of PRBs available at the BS. Then, the interplay amongst IoT devices, FAPs and PRBs can be formally characterized by defining the association within any triplet $(I_i, F_k, n) \in \mathbb{I} \times \mathbb{F} \times \mathbb{N}$ as

$$y_{k,i}^{n} = \begin{cases} 1, & \text{if the } n^{th} \text{ PRB is assigned to } (I_i, F_k) \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Interference can occur whenever the following scenarios happen: (i) the same PRB is assigned to two IoT devices I_i and I_j which are within the coverage area of F_k, or (ii) the same PRB is assigned to two IoT devices, wherein one of them (say I_i) is in coverage areas of both F_k and $F_{k'}$ and the other (say I_j) is in the non-overlapping regions of either F_k or $F_{k'}$.

Scenario (i) can be avoided if we ensure that no two IoT devices in Φ_k are allocated the same PRB, for any $k \in \mathbb{F}$. In other words,

(C_1) Given n^{th} PRB, we let $\sum_{i \in \Phi_k} y_{k,i}^{n} = 1$, for all $F_k \in \mathbb{F}$.

Similarly, Scenario (ii) can be avoided if we ensure that a given PRB is assigned only to one of the interfering IoT-FAP pairs, i.e.,

(C_2) Given n^{th} PRB, we need to ensure $y_{k,i}^{n} + y_{k',j}^{n} \leq 1$, for all $I_i \in \Phi_k \cap \Phi_{k'}$ and $I_j \in \Phi_k - \Phi_{k'}$, for any F_k and $F_{k'}$ in $\mathbb{F}$.

In most real-world settings, IoT devices generate tasks with deadlines in IoT-enabled 5G networks which have FAPs with non-identical computational capabilities in terms of CPU cycles. In such a scenario, task completion is typically affected by various factors such as transmission latency and execution time. In this paper, we assume that FAPs always execute and deliver all assigned jobs back to the respective IoT devices within the task deadline.

The above assumption is valid in the sense that BS leases the required number of PRBs to radio-enabled FAPs, and FAPs release the allocated PRBs upon successful delivery of tasks to respective IoT devices [16]. Furthermore, as time progresses, the infrastructure at the FAPs (e.g. GPU cards) can be augmented with better computational resources, making them more effective in delivering finished tasks within the prescribed deadline.

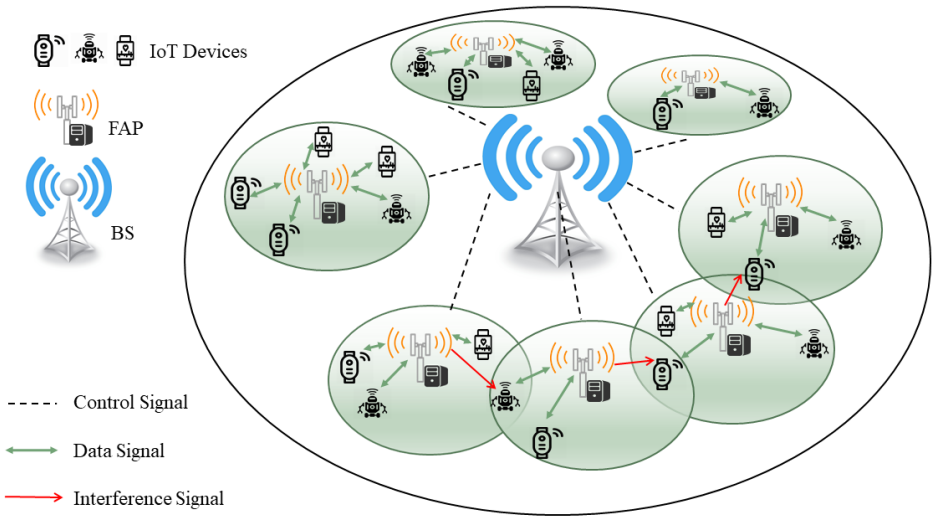

Figure 1: Communication framework for IoT enabled 5G.

4. Problem Formulation

Let $x_{k,i}$ denote the BS's match between the IoT device I_i with the FAP F_k, i.e.,

$$x_{k,i}(\boldsymbol{y}_{k,i}) = \begin{cases} 1, & \text{if } \sum_{n\in\mathbb{N}} y^n_{k,i} \neq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The goal of the BS is to maximize the task throughput, i.e., the total number of tasks executed in the network, which is given by,

$$\eta(Y) = \sum_{k\in\mathbb{F}} \sum_{i\in\mathbb{I}} x_{k,i}(\boldsymbol{y}_{k,i}), \tag{3}$$

via finding appropriate PRBs to IoT-FAP pairs in order to increase the overall productivity in terms of resource utilization. In this paper, we dismiss the problem of information fusion which arises whenever a given task is assigned to more than one FAP. In other words, we assume that

$$\sum_{k\in\mathbb{F}} x_{k,i}(\boldsymbol{y}_{k,i}) \leq 1 \tag{4}$$

holds true for each $I_i \in \mathbb{I}$. On the contrary, we let multiple tasks be assigned to a single FAP since these access points are capable of processing multiple tasks simultaneously.

In this paper, we assume that the BS leases the subset of PRBs $\mathbb{D}_i \in \mathbb{N}$ to the pair (I_i, F_k) until the completion of entire process. Furthermore, note that the BS prioritizes its assignment based on the priority values augmented by the IoT devices to their task requests. We denote the priority value declared by I_i regarding its task s_i using a binary variable $w_i \in \{0, 1\}$, where $w_i = 0$ corresponds to the lower priority weight and $w_i = 1$ corresponds to the higher priority weight. Let d_i denote the maximum number of PRBs

Table 1: Resource provisioning [17]

Priority Level	Tasks	Required No. of PRBs
High priority	Real-time gaming	5∼13
	Live streaming	5∼24
	IP multimedia signaling	5∼13
Low priority	Conventional video	3∼24
	File sharing	1∼110
	Web	1∼110

needed for successful completion of the task s_i. In order to satiate all high-priority tasks in the network, we assume

$$|\mathbb{D}_i| = d_i, \text{ for all } I_i \in \mathbb{I} \text{ such that } w_i = 1.$$

However, BS may allocate at most d number of PRBs for other tasks, i.e.,

$$|\mathbb{D}_i| \leq d_i, \text{ for all } I_i \in \mathbb{I}.$$

Moreover, Table 1 shows a relation between different tasks and required number of resources in a variety of real-time applications in wireless communication networks [17].

In summary, the resource allocation problem at the BS can be stated formally as follows:

$$\begin{aligned}
\max_{Y} \quad & \eta(Y) \\
\text{subject to} \quad & 1.\ \sum_{k\in\mathbb{F}} x_{k,i}(\boldsymbol{y}_{k,i}) \leq 1, \\
& 2.\ \sum_{i\in\Phi_k} y^n_{k,i} = 1, \text{ for all } F_k \in \mathbb{F}. \\
& 3.\ y^n_{k,i} + y^n_{k',j} \leq 1, \text{for all } I_i \in \Phi_k \cap \Phi_{k'} \\
& \qquad\qquad \text{and } I_j \in \Phi_k - \Phi_{k'}, \\
& 4.\ y^n_{k,i} \in \{0,1\}, x_{k,i} \in \{0,1\}, \\
& 5.\ \sum_{n\in\mathbb{N}} w_i y^n_{k,i} = d_i, \text{ for all } I_i \in \mathbb{I}, \\
& 6.\ \sum_{n\in\mathbb{N}} y^n_{k,i} \leq d_i, \text{ for all } I_i \in \mathbb{I}.
\end{aligned} \tag{P1}$$

Note that Problem P1 is a non-linear binary program, which is NP-hard in general [18]. However, Condition 3 enables us to construct an interference graph, which the BS can potentially use to distribute

PRBs across different FAPs without causing interference. Assuming that each PRB is represented using a unique color, we propose an efficient two-stage algorithm to solve Problem P1, where multiple colors are assigned to each FAP on the interference graph in Stage I, which are then distributed in Stage II to maximally improve task throughput. Moreover, given different constraints such as interference, varying resource demands, limited available PRBs, and a large number of IoT devices within densely deployed nonuniform FAPs, the proposed deterministic offline 2-stage algorithms satisfy these constraints yet coming up with a computationally feasible and near-optimal solution.

5. Efficient Two-Stage Algorithm

As pointed out in Section 3, interference can arise because of two reasons: Conditions C_1 (Constraint 2 in Problem P1) and C_2 (Constraint 3 in Problem P1). Note that Condition C_2 (Constraint 3 in Problem P1) unveils the interference caused by multiple FAPs to each IoT device. This condition can be summed up graphically using an *interference graph* $G \triangleq (V, E)$, (as shown in Stage 0 and Stage 1 of Fig. 2), where $V = \mathbb{F}$ represents set of FAPs, and an edge $e_{k,k'}$ lies in the edge set E if there exists at least one IoT device in the network which is interfered by the FAPs F_k and $F_{k'}$. In other words, we have

$$e_{k,k'} = \begin{cases} 1, & \text{if } F_k \text{ is a neighbor of } F_{k'} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

In this paper, we consider the multi-coloring problem on the *interference graph*, where interference relations are symmetric, i.e., $e_{k,k'} = e_{k',k}$, as shown in Stage 1 of Fig. 2. Such an assumption holds true when the transmissions are isotropic in nature. However, in the real world, interference relations can be asymmetric due to the presence of non-identical reflections at different locations. This problem is out of scope of this paper, and will be considered in our future work. Moreover, the asymmetric interference model may be handled by assigning the appropriate power level at each FAPs along with PRBs allocation.

Note that Constraint 1 ensures that any given task is assigned to only one FAP. We can avoid duplicate task assignments across FAPs via disintegrating Problem P1 into two stages: (i) we find the total number of PRBs assigned to each FAP in the first stage, and (ii) we distribute these assigned PRBs across different tasks at each FAP without any duplication. If we assume that each PRB can be represented with a unique color, the first stage reduces to *multi-coloring problem on the interference graphs*, and the second stage reduces to *distribution of colors* across different IoT devices. We use terms PRB and color interchangeably henceforth in this paper.

5.1. Stage I: Multi-Coloring Problem on Interference Graph

In solving a multi-coloring problem on the interference graph, the priority of the i^{th} task plays a critical role in the number of colors needed to solve Problem P1. Since Constraint 5 in Problem P1 is strict, we

need at most

$$\sum_{i\in\mathbb{I}}\left[w_i\cdot\left(\sum_{n\in\mathbb{N}}y_{k,i}^n\right)\right] \;=\; D_k^{min}\triangleq\sum_{i\in\Phi_k}w_i d_i \tag{6}$$

colors (PRBs) to serve all the high-priority tasks. On the other hand, we can evaluate the maximum number of colors (PRBs) needed to serve all the tasks by considering both Constraints 5 and 6 in Problem P1 as shown below.

$$\sum_{i\in\mathbb{I}}\sum_{n\in\mathbb{N}}y_{k,i}^n \;\leq\; D_k^{max}\triangleq\sum_{i\in\Phi_k}d_i. \tag{7}$$

In other words, if we denote the total number of colors (PRBs) assigned to F_k as

$$z_k\triangleq\sum_{i\in\mathbb{I}}\sum_{n\in\mathbb{N}}y_{k,i}^n, \tag{8}$$

then any feasible solution satisfies the inequality

$$D_k^{min}\leq z_k\leq D_k^{max}, \tag{9}$$

where D_k^{min} is the minimum number of colors needed to serve all high-priority tasks at F_k. In order to maximize spectrum efficiency, the BS can reuse colors across FAPs whenever it is possible.

Although several efficient algorithms have been proposed in the literature to solve multi-coloring problems on graphs, the constraints in our problem setting make our setting further challenging. As a result, we propose a novel algorithm in Section 6 to address this challenging problem efficiently.

5.2. Stage II: Distribution of Colors

In the next stage, the goal is to distribute the assigned colors to FAPs amongst various tasks in such a way that the total number of tasks served is maximized. The priority of the i^{th} task plays a critical role in the distribution of z_k colors available at the k^{th} FAP on the interference graph, at the end of Stage I. In other words, our goal in this stage is to maximize task throughput $\eta(Y)$, subject to the color assignment obtained from Stage I. More formally, we wish to solve the following problem.

$$\begin{aligned}
\max_{Y}\quad & \eta(Y)\\
\text{subject to}\quad & 1.\ \sum_{n\in\mathbb{N}}\sum_{i\in\mathbb{I}}y_{k,i}^n=z_k,\ \text{for all } F_k\in\mathbb{F},\\
& 2.\ \sum_{i\in\Phi_k}y_{k,i}^n=1,\ \text{for all } F_k\in\mathbb{F},\\
& 3.\ y_{k,i}^n\in\{0,1\},x_{k,i}\in\{0,1\}\\
& 4.\ \sum_{n\in\mathbb{N}}w_i y_{k,i}^n=d_i,\ \text{for all } i\in\mathbb{I},\\
& 5.\ \sum_{n\in\mathbb{N}}y_{k,i}^n\leq d_i,\ \text{for all } i\in\mathbb{I}.
\end{aligned} \tag{P2}$$

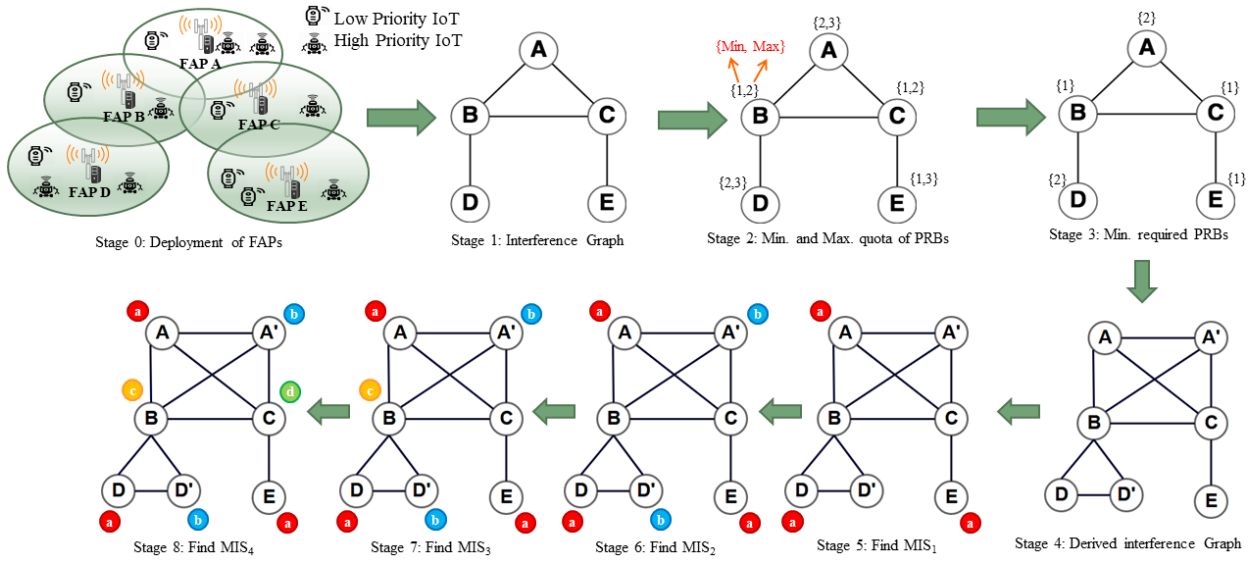

Figure 2: Illustration of Algorithm 1.

We propose our algorithmic solution to Problem P2 in Section 7.

6. Stage I: Multi-Coloring on Interference Graph

In this paper, our goal is to solve the multi-coloring problem on interference graphs with N (or fewer) colors under the constraint $D_k^{min} \leq z_k \leq D_k^{max}$. Note that multicoloring problem in the unconstrained setting itself is NP-Hard in general. Therefore, we propose a greedy algorithm to solve the multicoloring problem via picking maximal independent sets on residual graphs in an iterative manner.

However, the availability of limited number of colors in our problem setting introduces a new challenge in terms of the feasibility of the multicoloring solution. Therefore, we first investigate whether there exists a feasible solution to Stage I. In the special case where colors are not reused, the minimum number of colors needed to solve the multi-coloring problem on a given interference graph is given by $\sum_{k=1}^{K} D_k^{min}$. However, since we allow reuse of colors to improve spectrum efficiency, the minimum number of colors, denoted as the set L, required to solve the multi-coloring problem on a given interference graph is feasible only if $|L| \leq N$. In this paper, we find the set L and its complement $R = \mathbb{N} - L$ (number of colors available for serving low-priority tasks) using Algorithm 1.

6.1. Minimum Resource Requirement

The problem of finding the minimum number of required PRBs at FAPs is not trivial. For instance, consider 10 non-interfering FAPs in the network, and the minimum required PRBs at each FAP is two. Instead of assigning 20 PRBs, we can just assign two PRBs to each FAP and fulfill the minimum requirement

Algorithm 1 Minimum Resource Requirement Algorithm

Input: Minimum required resources $D_k^{min}, \forall F_k \in \mathbb{F}$, interference graph G, FAP set $\mathbb{F}$, PRB set $\mathbb{N}$, a set $L := \phi$, count=0.

Output: Number of resources R for elaborated FAPs.

1: **for** all node k in G do
2: Create clique of size D_k^{min}.
3: Each node of clique D_k^{min} inherits all the edges of $k \in G$ and a new extended interference graph $G'(V', E')$ is generated corresponding to steady FAPs requirements.
4: **end for**
5: **while** all nodes are not assigned with a color or count $\leq |\mathbb{N}|$ do
6: Select an MIS from V'
7: Assign the lowest index color $n \in \mathbb{N}$, which is not already assigned to any node
8: $L = L \cup \{n\}$, count++
9: **end while**
10: $R = \mathbb{N} - L$

at all the 10 FAPs since there is no interference between them. On the other hand, if the interference graph between the same 10 FAPs is a clique, then we need 20 PRBs to satisfy the minimum requirement at all the FAPs. In other words, diverse interference patterns can result in a wide variation in the minimum number of PRBs required to serve only high-priority tasks.

In order to find the minimum assignment for the high-priority tasks, we construct a new graph $G'(V', E')$ via replacing the FAP F_k with a virtual clique of size D^{min}, where each node in this clique inherits the interference relations of F_k from the original graph G, for all $F_k \in V$. Since κ-coloring problem is NP-complete [19], we apply a greedy graph-coloring technique to find out the minimum number of required PRBs. We select a Maximal Independent Set (MIS) from the set V', and assign a color with the smallest index which has not been assigned previously (lines 5-7). The assigned color is updated in set L (line 8). The above process continues for all vertices in V'. Now the set L contains all required colors (i.e., PRBs) for steady FAPs, and R is the remaining set of colors that can be assigned to the elaborated FAPs set.

For the sake of illustrating Algorithm 1, consider the following example with the set of FAPs and PRBs defined as $\mathbb{F} = \{A, B, C, D, E\}$ and $\mathbb{N} = \{a, b, c, d, e, f, g\}$ respectively. We have shown the deployment of FAPs and respective interference topology in Stage 0 and Stage 1 of Fig. 2. For the sake of simplicity we have assumed $d_i = 1$ for all IoT devices in the system. The minimum and maximum requirement of PRBs at each node are shown in Stage 2. Stage 3 shows scenario of minimum requirement of PRBs at each node. Furthermore, Stage 4 shows a derived interference graph where each node is extended to minimum required PRBs and edges are formed based on line 2-3 of Algorithm 1. Nodes A, D and E are selected as an MIS

Algorithm 2 Interference Graph Transformation Algorithm

Input: Quota of resources $D_k^{min}, D_k^{max} \ \forall F_k \in \mathbb{F}$, interference graph $G(V, E)$, PRB set $\mathbb{N}$

Output: FAP set $\breve{\mathbb{F}}$, PRB set $\breve{\mathbb{N}}$, maximum quota of FAPs $\breve{D}_k^{max}, \forall k \in \breve{\mathbb{F}}$, derived interference graph $\breve{G}(\breve{V}, \breve{E})$.

1: $\breve{\mathbb{F}}^s = \phi, \breve{\mathbb{F}}^e = \phi, \breve{\mathbb{N}} = \mathbb{N}$.

2: **for** all $F_k \in \mathbb{F}$ do

3: $\quad \breve{\mathbb{F}}^s = \breve{\mathbb{F}}^s \cup \{F_{k^s}\}, \breve{\mathbb{F}}^e = \breve{\mathbb{F}}^e \cup \{F_{k^e}\}$

4: $\quad D_{k^s}^{max} = D_k^{min}$, $D_{k^e}^{max} = D_k^{max} - D_k^{min}$.

5: **end for**

6: **for** all $k \in V$ do

7: $\quad$ Add k^s and k^e to vertex set $\breve{V}$ of $\breve{G}(\breve{V}, \breve{E})$.

8: $\quad$ k^s and k^e inherits all the edges of k in G.

9: $\quad$ Create an edge between k^e, k^s and update edge set $\breve{E}$ of $\breve{G}(\breve{V}, \breve{E})$.

10: **end for**

and assigned with the lowest available color a based on lines 6-7. Now set $L = \{a\}$. Accordingly, lines 5-9 executed for remaining uncolored nodes. Stage 8 shows the final stage where each node is assigned with a valid color and set $L = \{a, b, c, d\}$. Thus, $R = \mathbb{N} - L$, i.e., $R = \{e, f, g\}$ can be assigned to elaborated nodes in-order to fulfill the maximum quota of respective FAPs.

6.2. Interference Graph Transformation

Before we distribute the colors to FAPs on the interference graph, it is necessary to pre-process the interference graph structure to accommodate both high-priority and low-priority task requests simultaneously. In this regard, Algorithm 2 transforms an interference graph G into a *derived interference graph* $\breve{G}$, where each FAP in G is split into two dummy FAPs called the *steady* FAP and *elaborate* FAP, as stated in Line 3 of Algorithm 2. The set of steady FAPs $\breve{\mathbb{F}}^s$ keeps track of minimum requirement at the respective FAP, whereas the set of elaborated FAPs $\breve{\mathbb{F}}^e$ handles the remaining quota, i.e., the difference between the maximum and the minimum requirements which are allocated to serve low priority tasks. Thus, the number of colors needed at the steady and elaborated FAPs are respectively given as

$$D_{k^s}^{max} = D_k^{min} \quad \text{and} \quad D_{k^e}^{max} = D_k^{max} - D_k^{min},$$

as computed in Line 4 in Algorithm 2.

Since $\breve{\mathbb{F}}^s$ and $\breve{\mathbb{F}}^e$ are virtual representations of $\mathbb{F}$, the derived interference graph $\breve{G}$ inherits all the interference relations from G. In other words. if F_{k^s} and F_{k^e} represent the *steady* and *elaborated* FAP of original FAP F_k respectively, then both F_{k^s} and F_{k^e} inherit the edges of F_k from G (lines 6-10). We use the transformed graph as an input to the resource allocation algorithm in Section 6.3.

Algorithm 3 Resource Allocation Algorithm

Input: Maximum quota of FAPs $\breve{D}_k^{max}, \forall k \in \breve{\mathbb{F}}$, transformed interference graph $\breve{G}(\breve{V}, \breve{E})$, and R

Output: PRBs allocated nodes, i.e., $\breve{\mu}(v_k)$, $\forall v_k \in V$.

1: $\forall k \in \breve{\mathbb{F}}, \breve{\mu}(k) = \phi$, waiting list $W_k = \phi$.
2: $\forall n \in \breve{\mathbb{N}}, \breve{\mu}(n) = \phi$, candidate list $\Re_n = \breve{\mathbb{F}}$.
3: **while** $\exists \Re_n \neq \phi$ do
4: **for** all PRB n with $|\Re_n| > 0$ do
5: $Z_n := \forall k \in \Re_n, \forall k' \in \breve{\mu}(n), e_{k,k'} = 0$.
6: Find an MWIS on Z_n as Z_n^{max}.
7: **end for**
8: **if** $\forall n, Z_n^{max} = \phi$ then
9: Return $\breve{\mu}(k)$
10: **else**
11: **for** all FAP $k \in Z_n^{max}$ do
12: BS sends $msg < k, n >$
13: $\Re_n = \Re_n - \{k\}$
14: **end for**
15: Upon receive $msg < k, n >$ on FAP k
16: FAP k updates its waiting list $W_k = W_k \cup \{n\}$.
17: **end if**

18: **for** all FAP k for which $W_k \neq \phi$ do
19: **if** $k \in \breve{\mathbb{F}}^s$ then
20: k accepts $D_{k^s}^{max}$ PRBs from $W_k \cup \breve{\mu}(k)$ as new $\breve{\mu}(k)$ and $W_k = \phi$.
21: **else**
22: $count = 0$, Index $l = 1$
23: **for** all $k \in \breve{\mathbb{F}}^e$ do
24: $W_k = W_k \cup \breve{\mu}(k), \breve{\mu}(k) = \phi$
25: **end for**
26: **while** $count < |R|$ and $\exists k^e, W_{k^e} \neq \phi, |\breve{\mu}(k^e)| < \breve{D}_{k^e}^{max}$ do
27: **if** $|\breve{\mu}(k_l^e)| < \breve{D}_{k_l^e}$ and $W_{k_l^e} \neq \phi$, then
28: FAP k_l^e picks n from $W_{k_l^e}$, s.t., $\breve{\mu}(k_l^e) = \breve{\mu}(k_l^e) \cup n$.
29: $W_{k_l^e} = W_{k_l^e} - \{n\}$
30: $count = count + 1$
31: **end if**
32: $l = l + 1$
33: **end while**
34: **end if**
35: $W_k = \phi$
36: **end for**
37: BS updates allocations $\breve{\mu}(n) \leftarrow k$ and $\breve{\mu}(k) \leftarrow n$, for $\forall n \in \breve{\mathbb{N}}, k \in \breve{\mathbb{F}}$ based on above lines 18-36.
38: **end while**
39: Merge the respective steady and elaborated nodes, remove the self loop edges, replace the parallel edges with one edge. /*construct the original interference graph $G(V, E)$*/
40: $v_k = k^e \cup k^s, \breve{\mu}(v_k) = \breve{\mu}(k^e) \cup \breve{\mu}(k^s), \forall v_k \in V, k \in \breve{\mathbb{F}}$ /* Allocate the set of assigned colors of steady and elaborated nodes to merged node */.

6.3. Resource Allocation Algorithm

Given the derived interference graph $\breve{G}$ and resources $\mathbb{N}$, Algorithm 3 demonstrates how resources are distributed amongst steady and elaborated FAPs using two different allocation schemes in-order to fulfill the

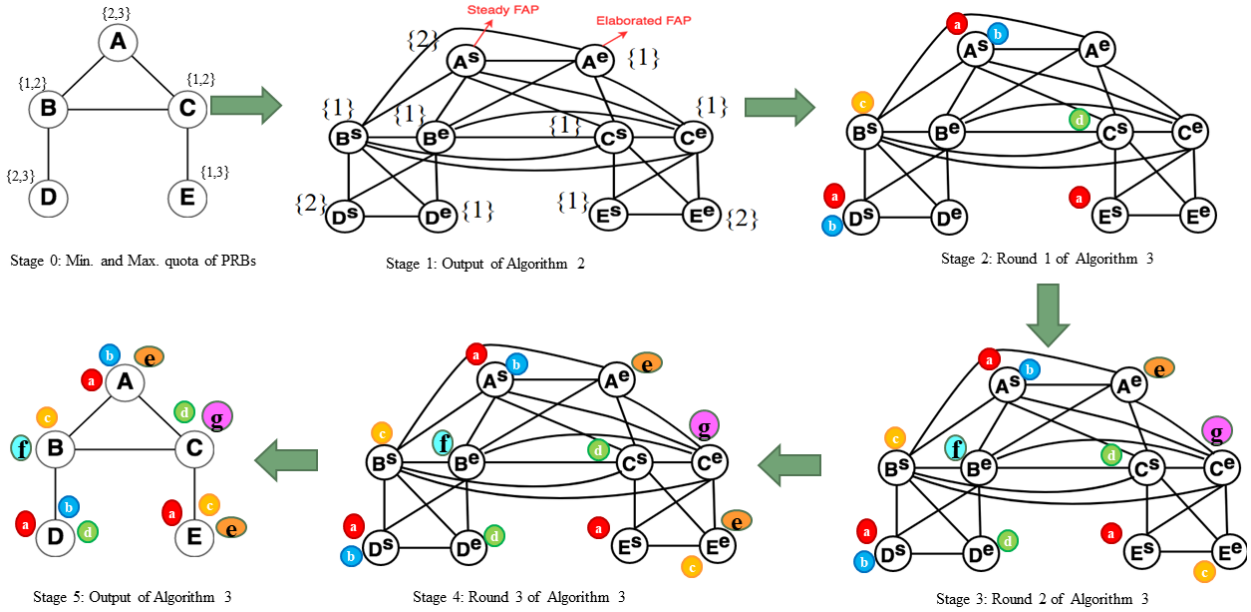

Figure 3: Illustration of Algorithm 2 and Algorithm 3.

minimum and maximum required PRBs. The first allocation scheme (Line 20 in Algorithm 3) allocates the colors needed to serve the high-priority tasks (minimum requirement), where the second allocation scheme (Lines 22-33) assigns the colors to cover the maximum demand to the greatest possible extent.

However, before we delve into the details of these two allocation schemes, Algorithm 3 first identifies potential matches between FAPs and colors via iteratively associating each color $n \in \mathbb{N}$ to a maximum weighted independent sets (MWIS) Z_n^{max}, as shown in Lines (4-6). For a non-empty Z_n^{max}, the BS sends a message msg (Line 12) that reveals a potential match between the color n and the FAP node F_k. Upon receiving this message, F_k includes the color n in its waiting list W_k (Line 15-16).

Once all the colors are exhausted, the allocation schemes decide if a given association is acceptable. In order to give higher priority to serve nodes in $\breve{\mathbb{F}}^s$, the first allocation scheme accepts $D_{k^s}^{max}$ colors at each F_{k^s}, and releases the unused colors in the waiting list W_{k^s} (Line 20). In the subsequent stage, the second allocation scheme gets activated where the elaborated nodes accept colors in their respective waiting lists in a sequential order until their respective demand $D_{k^e}^{max}$ is fulfilled, or all the colors have been exhaustively assigned (Lines 22-36). Lines 39-40 merges the assignments made to steady and elaborate nodes to compute the final assignment on the original interference graph G.

Figure 3 illustrates Algorithms 2 and 3, using the same example discussed in Figure 2. Based on Algorithm 2, the interference graph is converted into a derived interference graph along with respective PRBs demand at each vertex as shown in Stage 1 of Fig. 3. Based on lines 4-6 of Algorithm 3, let BS prepares the set of MWIS for each color as follows: $\Re_a = \{A^s, D^s, E^s\}$, $\Re_b = \{A^s, D^s\}$, $\Re_c = \{B^s, E^e\}$,

$\Re_d = \{C^s, D^e\}$, $\Re_e = \{A^e, E^e\}$, $\Re_f = \{B^e\}$, and $\Re_g = \{C^e\}$ and sends a message to each node in the sets. Upon receiving the message, each steady node F_{k^s} selects $D_{k^s}^{max}$ number of colors (PRBs) (based on lines 15-20 of Algorithm 3) in Stage 2 of Fig. 3. Based on lines 22-35, each elaborated node then selects a PRB from its waiting list as shown in Stage 3. Nodes A^e, B^e, C^e and D^e have received the number of demanded PRBs. Node E^e is still short of one more color, thus in the next iteration it selects color e from its waiting list. Thus, the final allocation shows a valid PRB assignment scenario and this allocation also fulfills the required number of PRBs at each FAP. However, Stage 5 is the final output after execution of lines 39-40 of Algorithm 3.

7. Stage II: Distribution of PRBs to IoT

Having identified the color assignment at the FAPs, we propose Algorithm 4 to redistribute the PRBs assigned to a given FAP to its respective IoT device (say I_i) based on their respective priority levels (w_i) and demands (d_i). Algorithm 4 allocates the PRBs to each IoT device using different schemes for higher priority (Lines 2-9) and lower-priority (Lines 10-17) IoT devices. FAP allocates the number of required PRBs to higher priority IoT devices as per their demand, update the set $\breve{\mu}(k)$ and variable $y_{k,i}^n$ (Lines 2-6). However, lower priority IoT devices are kept into a waiting list $\mathbb{W}_k$ (Line 7) before the final allocation. Elements of set $\mathbb{W}_k$ are sorted in increasing order based on the required number of PRBs (Line 10). For each lower priority IoT device in this sorted list, PRBs are assigned iteratively via updating the set $\breve{\mu}(k)$ until their demand is fulfilled or the colors are exhausted for the corresponding FAP (Lines 12-15).

We illustrate the Algorithm 4 in Fig. 4. Stage 0 shows the output of Algorithm 3 which is an input to Algorithm 4. Stage 1 of Fig. 4 represents the first round of Algorithm 4 where, FAP A assigned the available PRBs to higher priority IoT devices followed by lower priority IoT devices. In the second round FAP B assigns PRB (color) c to higher priority IoT device and then PRB f to lower priority IoT device. In the 3rd round FAP C assigns the PRBs d and g to higher priority and lower priority IoT devices respectively. In the 4th round PRBs a, b and d are assigned to two higher priority devices and one lower priority device. In the 5th round, PRB a is assigned to higher priority device. There are two unallocated lower priority IoT devices exist. These two IoT devices are sorted in ascending order based on PRBs demand (line 10 of Algorithm 4). However, as there demands are equal thus anyone can be assigned first. Hence, the remaining PRBs c and e are assigned to two lower priority devices. Moreover, the final allocation satisfied the allocation constraints and fulfilled the required number of PRBs at each IoT device.

8. Computational Complexity, Validity and Resource Utilization

In this section we demonstrate the theoretical analysis of our proposed algorithm as follows:

Theorem 1. *The time complexity of our proposed scheme is* $O(K(NK\breve{V}^3 + M \log M))$.

Algorithm 4 Distribution of PRBs to IoTs

Input: PRBs allocated FAPs, i.e., $\breve{\mu}(k)$, $\forall F_k \in \mathbb{F}$; d_i, w_i, $\forall I_i \in \mathbb{I}$

Output: PRBs allocated IoTs.

1: **for** all FAP $F_k \in \mathbb{F}$ do
2: **for** all IoT $I_i \in \Phi_k$.
3: **if** $w_i = 1$
4: Allocate d_i PRBs to IoT I_i
5: $y^n_{k,i} = 1$
6: $\breve{\mu}(k)$=$\breve{\mu}(k) \setminus d_i$ /*update the remaining PRBs*/
7: **else** $\mathbb{W}_k = \mathbb{W}_k \cup I_i$
8: **end if**
9: **end for**
10: Sort $I_i \in \mathbb{W}_k$ in increasing order of d_i
11: **while** $d_i \neq 0 \forall I_i \in \mathbb{W}_k$ or $\breve{\mu}(k) \neq 0$ **do**
12: Allocate PRB $n \in \breve{\mu}(k)$ to IoT I_i
13: $y^n_{k,i} = 1$
14: $d_i = d_i - 1$
15: $\breve{\mu}(k) = \breve{\mu}(k) \setminus n$
16: **end while**
17: **end for**

Proof. The time complexity of Algorithm 1 is $O(|V'|^3)$. The interference graph transformation procedure given in Algorithm 2 takes $O(K + N)$ computational complexity. In order to estimate the computational complexity of Algorithm 3, we first compute the computational complexity of steady FAPs then for elaborated FAPs. Whenever the BS sends message to an FAP, it removes that FAP from its candidate list (as shown in line 4 of Algorithm 3). At the end, for each PRB (color), the set of interference free FAP assigned with the same PRB. As we have N PRBs, each time BS selects FAP by looking into the MWIS from the graph. However, finding an MWIS would take $O(\breve{E})$ time complexity in the worst-case scenario to visit all the edges in derived interference graph $\breve{G}(\breve{V}, \breve{E})$. However, this process has to be done for all nodes in the derived interference graph $\breve{G}(\breve{V}, \breve{E})$. Thus, the total time complexity could be $O(\breve{V}\breve{E})$, i.e., $O(\breve{V}^3)$ using the above greedy approach. Moreover, the number of MWIS is bounded by the total available PRBs, i.e., 100 in the network. Thus, approximate algorithm for MWIS will take $O(\breve{V}^3)$ time complexity in the worst case scenario using the greedy approach and set of network constraints. Thus, the computational time complexity to allocate a valid PRB to each steady FAP be $O(KN\breve{V}^3)$. When the Algorithm 3 assigns the resource to an elaborated FAP (lines 26-36), all the elaborated FAPs need to be traversed and that

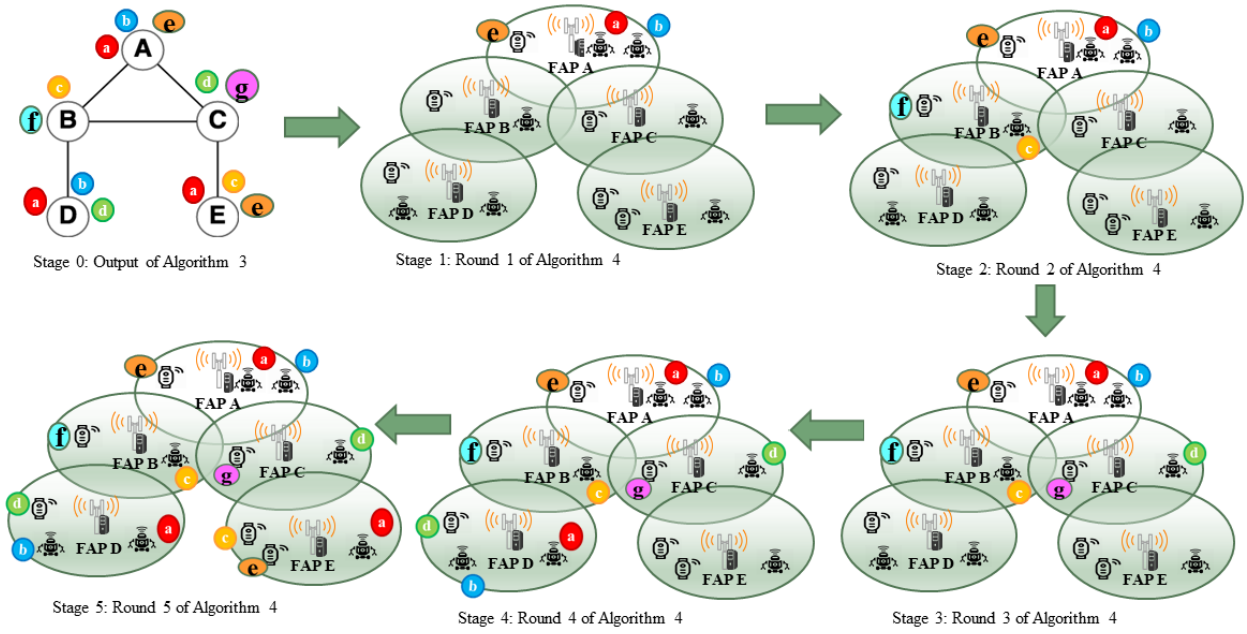

Figure 4: Illustration of Algorithm 4.

may take $O(K)$ computational time complexity. Thus, the computational time complexity of lines 1-38 of Algorithm 3 becomes $O(NK^2\breve{V}^3)$. However, time complexity of lines 39-40 is $O(\breve{\mathbb{F}})$. Computational time complexity of Algorithm 4 is $O(KM \log M)$ to distribute the resources at the IoT level. The time complexity of our proposed algorithm is primarily dependent on Algorithm 3. Thus, the overall time complexity of our proposed algorithm is $O(NK^2\breve{V}^3 + KM \log M)$. □

8.1. Performance Bound and Validity of Proposed Algorithm

We compute the PRB bound over the propagation latency of our proposed scheme. Let $\mathbb{H}$ and $\mathbb{L}$ be the number of higher priority and lower priority IoT devices associated with each FAP, respectively. Assuming that each IoT device has to send 504 bit of data within a propagation delay of 0.5 ms and BS operates at 64 QAM, then there is a need of at least one PRB for an IoT device [20], [21]. If Q and Δ represent clique and degree of interference graph $G(V, E)$ respectively, then we bound the number of required PRBs in the network as follows:

Lemma 1. *If propagation latency of IoT devices carrying 504 bits of data is bounded by 0.5 ms, then the allocated number of PRBs* Γ *in the network is bounded by* $\mathbb{H}Q \leq |\mathbb{N}| \leq (\mathbb{H} + \mathbb{L})(\Delta + 1)$.

Proof. As the interference graph has a clique of size Q and each node has minimum PRB demand of $\mathbb{H}$, consequently to fulfill the minimum PRB demand of each FAP, the number of required PRB $|\mathbb{N}|$ must follow the lower bound constraint such as $Q\mathbb{H} \leq |\mathbb{N}|$. On the other hand, based on graph coloring, the maximum

number of required colors is given by $(\Delta + 1)$, which occurs when each node is labeled with a single color [22]. Since each node is assigned with a maximum of $(\mathbb{H} + \mathbb{L})$ number of PRBs, the required number of PRBs is therefore upper-bounded by $(\mathbb{H} + \mathbb{L})(\Delta + 1)$. □

Lemma 2. *The proposed algorithm results in a valid PRB assignment.*

Proof. To prove the validity, we need to justify the following three conditions: (i) minimum requirement of FAPs is fulfilled, (ii) assignment is interference-free, (iii) higher priority IoT devices received their required number of PRBs. The successful execution of proposed Algorithm 1 ensures that the minimum requirement of FAPs is to be fulfilled. Algorithm 2 and Algorithm 3 provide the interference-free assignment of PRBs. The derived interference graph (an outcome of Algorithm 2) ensures the interference constraint by inheriting the respective edges from the original interference graph. Moreover, BS selects MWIS set from the derived interference graph and sends a message to each node in the set for n^{th} PRB. However, BS selects unique PRB for each set, which avoids the allocation of the same PRB to any two nodes within a one-hop neighbor in the derived graph (Lines 4-7 of Algorithm 3). Moreover, each steady node selects the required number of PRBs from the waiting set. This phenomenon ensures that each FAP is assigned with the minimum required PRBs. Furthermore, elaborated nodes select the remaining PRBs from set R in order to satisfy the maximum quota of each FAP. Algorithm 4 ensures the allocation of PRBs to higher priority IoT devices (Lines 3-6). However, the remaining PRBs are further distributed to lower priority IoT devices (Lines 10-15 of Algorithm 4). Thus, the proposed algorithm follows the above three points. Hence, the final allocation results in a valid PRB assignment. □

9. Performance Study

In this section, we evaluate our proposed method based on the following environments. We have assumed that the set of FAPs and IoT devices are randomly deployed in a cellular network underlying BS. The operational bandwidth of BS is considered as 20 MHz accordingly, the maximum number of available PRBs is found as 100 [15]. We considered that a PRB can carry 504 bits of data at 64 QAM modulation scheme [14]. The network model is considered the same as [10]. The required latency, data size, and corresponding CPU cycles are determined by specific device types. We have considered CPU frequency of FAP processor as 1.4 GHz and computational complexity of the task as 10 computation cycles/bit. [12]. We have assumed that co-CPU tasks share the equal CPU rate of respective FAP. An IoT device generates a single task at a time to get executed at an FAP.

In the performance study, we have considered two evaluation schemes for computing Utility and Task Throughput to analyze the proposed algorithms.

9.1. Task Throughput

The performance of the proposed algorithms (1-4) has been evaluated and analyzed through a simulated experiment with the goal to maximize task throughput ($\eta(Y)$) in compliance with the pre-determined constraints formulated in the previous sections. The evaluations are carried out in terms of number of available resources (PRBs), interference link density, number of Fog Access Points (FAPs), number of IoT devices and their respective maximum demand.

The simulated network topology consists a maximum number of I= 500 IoT devices each equipped with a demand $d_i \in [1, 5]$.The priority of each IoT device is randomly chosen with an equal probability of high and low, setting d_i to 1 or 0 for high and low priority respectively. In our results, we assume that several tasks can be generated by an IoT device, where each IoT device may need more than one PRB to get executed. Assume a number of F FAPs, $F \in [10, 100]$ where all FAPs are randomly distributed in the network. Each IoT device is also randomly assigned to FAPs in the network. It is worth mentioning that with edge allocation in the simulated environment is achieved by edge assignment based on distance between the interfering FAPs. In other words, if the computed Euclidean distance between the two FAPs is bounded within a threshold communication range (R), implying that the FAPs are interfering, hence there is an edge assignment between the FAP nodes. As we increase the communication range between FAPs, the number of links in the interference graph increases resulting in higher graph density. The communication range between FAPs for an interference graph construction is set as R=50 m and the deployment area of the network is set to 200m $\times$ 200m.The simulation parameters are shown in Table-1. All algorithms are implemented in Python language utilizing open-source frameworks of Networkx and Matlplotlib. For each simulation scenario, the performance results are averaged over 50 iterations.

The simulated experiments are conducted in two sections *a)* When IoT devices generates a single task with a demand = 1 *b)* When IoT devices generate more than one task with demand $d_i \in [1, 5]$.

9.1.1. IoT devices d_i = 1

When the maximum IoT demand is set to 1, it implies that a task generated by an IoT device will need only one PRB to get it executed. Therefore, the total number of tasks served by our proposed algorithm is equal to the total number of PRBs assigned by the FAP across the network. The impact of different network parameters on task throughput is shown as follows:

A) Impact of Graph Density on Task Throughput: As discussed earlier, $\eta(Y)$ represents the maximum number of tasks executed in the network. Comparison of task throughput with respect to (w.r.t) network density is shown in Fig.5. In the given figure, the number of FAPs and the number of IoTs are assigned to a value of 20 and 100 respectively in the network. From the results, it is evident that as the graph density increases linearly, the $\eta(Y)$ value decreases linearly. This trend in the graph is attributed to the fact that with stronger interference between the FAPs, resource res-usability decreases. For a graph density close to

Table 2: Simulation Settings.

Parameters	**Values**
Maximum number of available PRBs	100
Range for each IoT demand (d_i)	$[1, 5]$
Maximum number of IoT devices	500
Maximum number of FAPs	250
Range for interference link density	$[0.0, 1.0]$
Interference range	50 (m)
Deployment area	200 (m) x 200 (m)
CPU processing rate	1.4 GHz
Task computational complexity	10 cycles per bit
Communication data rate	504 bits per PRB per time unit
Priority of IoT device (i^{th})	{1,0}

0, the $\eta(Y)$ value is comparable to the number of IoTs in the network which is 100 as all the IoT devices (high and low priority) are serviced. Alternatively, for a given graph density as the number of available resources (PRBs) increase from 10 to 50, the $\eta(Y)$ value is higher as with increased number of PRBs more IoT demands are being serviced. Thus, as the graph density increases there is increased interference between the FAPs in the network, which deters the resource reusability resulting in a decreased value for $\eta(Y)$.

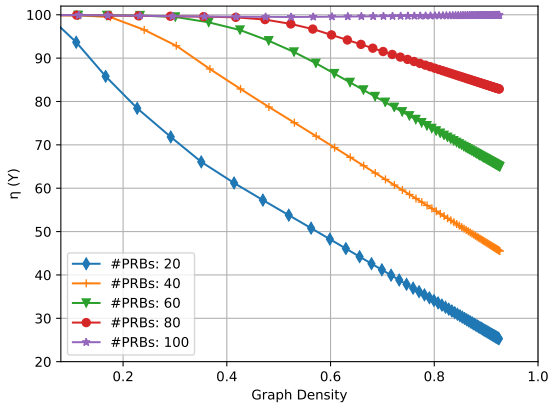

Figure 5: $\eta(Y)$ vs. graph density.

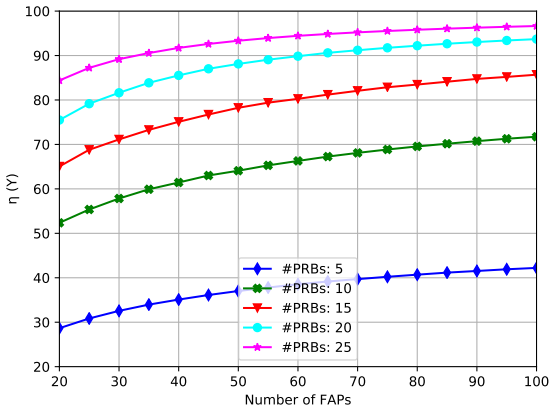

Figure 6: $\eta(Y)$ vs. number of FAPs.

B) Impact of Number of FAPs on Task Throughput: Fig. 6 demonstrates the relationship between the number of FAPs and $\eta(Y)$ for a given number of IoT devices as 100. It is worth mentioning that there is a

random assignment of IoT devices across FAP nodes and with the increase in FAP nodes the total network demand aggregated from the IoT devices remains the same. Thus, as the number of FAPs increases there is a better distribution of IoT devices across these FAPs. Hence, there is a better utilization of resources across the network with the proposed re-usability scheme which leads to a higher task throughput value, $\eta(Y)$.

C) Impact of Available Resources on Task Throughput: Fig.7 represents the co-relation between available PRBs in the network and task throughput, $\eta(Y)$. For a given number of IoTs as the number of PRBs increase linearly from 20 to 100, the $\eta(Y)$ value increases until it starts converging to the number of IoTs implying that the demand for all IoT devices is met. Alternatively, for a given number of available resources (PRBs), as the number of IoT devices increase from 20 to 100, there is an increased demand for resources in the network and hence, more number of IoT devices are serviced with the given PRBs resulting in higher value of $\eta(Y)$. Fig.7 also highlights the $\eta(Y)$ value where the aggregated demand for all high priority IoTs is met using a dot on each curve, i.e. where D_k^{min} for all FAPs is satisfied. Beyond this point, the PRBs are allocated to fulfill the demand for all low priority IoTs to the maximum extent. Fig.8 represents the same comparison for $\eta(Y)$ w.r.t available PRBs by varying the number of IoTs from 100 to 500. For the number of IoT devices set to 500, the $\eta(Y)$ value is higher and tends towards 500 while servicing the demand for 300 IoT devices with an availability of 100 PRBs in the network. Another important observation in this graph is that while it services 300 IoT devices, it has satisfied the demand for all the high priority IoT devices represented with a dot on the curve. This applies to other curves as well for number of IoTs in $[100, 400]$ where requests for all 100% of high priority IoT devices is fulfilled.

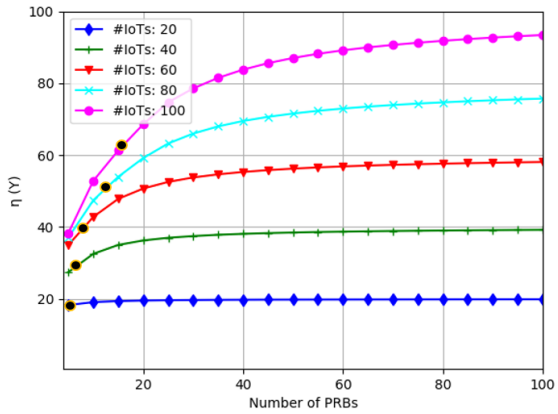

Figure 7: $\eta(Y)$ vs. number of PRBs, IoT=100.

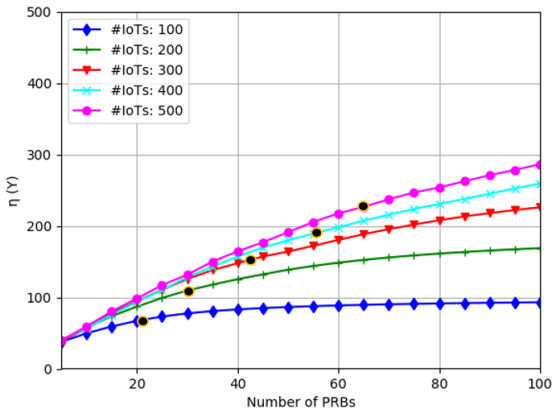

Figure 8: $\eta(Y)$ vs. number of PRBs, IoT=500.

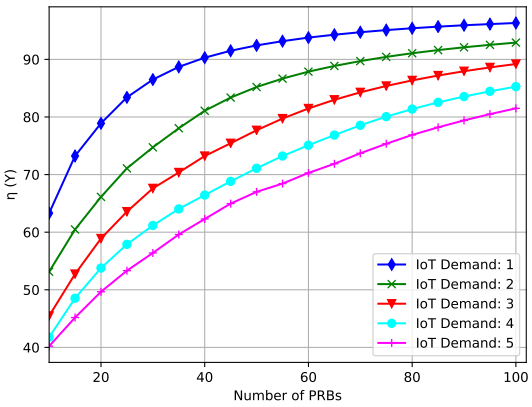

Figure 9: Impact of varying IoT's demand on $\eta(Y)$ w.r.t PRBs.

9.1.2. IoT devices $d_i \in [1, 5]$

In the following simulation scenario, we will assume that more than one PRBs are needed to service tasks generated by an IoT device. To achieve this, we have the IoT demand, d_i, set to a uniform random distribution in $[1, 5]$ for both high and low priority devices. Following use-cases will demonstrate the impact of varying IoT demand and network parameters on total task throughput of the network.

A) Impact of Varying IoT Demand on Task Throughput: For the given network topology, maximum IoT demand, d_i, is a uniform random distribution in $[1, 5]$, IoT devices set to 100 randomly distributed across a network of 20 FAPs. Fig.9 demonstrates how the maximum IoT demand for each IoT device in the network impacts the aggregated $\eta(Y)$. From the results we can observe that for a given number of PRBs as the IoT demand varies from 1 to 5, the total demand in the network increases resulting in a decreased task throughput ($\eta(Y)$) due to unavailability of resources to service the IoT requests. Alternatively, for a given upper bound on IoT demand as we increase the number available PRBs in the network, the $\eta(Y)$ value increases with gradual saturation towards the number of IoTs as more with increased resource availability more the demand for more number of IoT devices is being met.

B) Impact of Varying IoT Demand and PRBs on Task Throughput: Fig. 10 and 11 demonstrate the impact of varying FAPs on ($\eta(Y)$) when the maximum IoT demand is in $[1, 5]$. It is evident that for IoT count set to 20 and 40, as the number of available resource increase in the network, the $\eta(Y)$ also increases at a constant untill it reaches to a value close to the number of IoT devices, implying that all the IoT devices are being serviced. However, for an IoT count set above 40, as the number of PRBs increase approaching to a maximum limit of 100, the ($\eta(Y)$) value increases gradually. It tends towards the number of IoTs but it does not saturate closely to IoT count with maximum available PRBs as opposed to that in Fig. 7. This is because with increase in number of IoT demand for each device, more than one PRBs are being allocated

to IoT devices to fulfill the generated tasks while ensuring that the demand for high priority tasks is met first. In such a case,there are no resources left to meet the demand of the remaining volume of IoT devices resulting in lower $\eta(Y)$ value in contrast to Fig. 7. Similarly, Fig. 11 also exhibits the comparison of $\eta(Y)$ and PRBs with an IoT count set in $[100, 500]$, each with a varying demand. From the graph we can observe that for 500 IoTs, as the PRBs increase the task throughput function, $\eta(Y)$ increases very slowly thereby servicing the requests for only 150 IoT devices with a maximum PRBs of 100. This task throughput is roughly half the $\eta(Y)$ obtained in Fig. 8 where nearly 280 devices were being serviced for the same PRBs and IoT count.

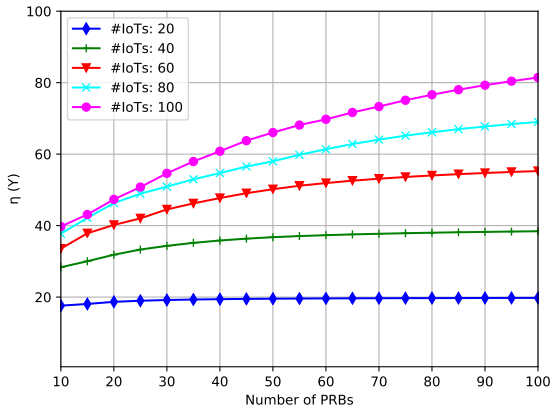

Figure 10: $\eta(Y)$ vs. PRBs, IoT=100, $d_i \in [1, 5]$

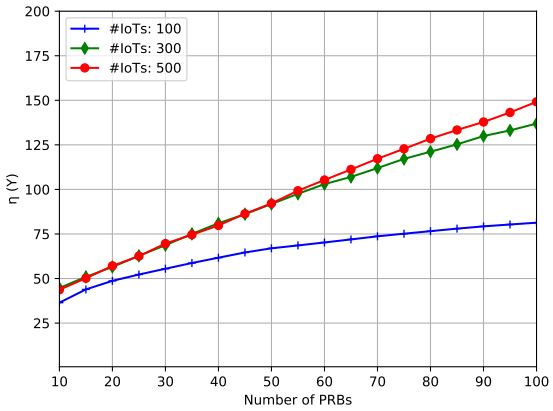

Figure 11: $\eta(Y)$ vs. PRBs, IoT=500, $d_i \in [1, 5]$

9.2. Utility

In addition to task throughput, we also evaluate the performance of the proposed algorithm by determining utility based resource allocation in IoT framework. In the field of network communication, utility is defined as a ratio of assigned PRBs to the total PRBs demanded by IoT devices. It is a monotonically increasing function of the total number of tasks served by the proposed algorithm. Mathematically, it can be represented as follows: $Utility_i = \frac{1}{d_i} \sum_{n \in \mathbb{N}} y^n_{k,i}$, for any $F_k \in \mathbb{F}$. We define the utility of higher priority and lower priority IoT devices as follows:

$$Utility_i = \begin{cases} 1, & \text{if } w_i = 1 \\ \leq 1, & \text{otherwise.} \end{cases} \tag{10}$$

A) Impact of IoT Devices on Utility: For an experimental analysis, a set of FAPs ranging from 50 to 250 was chosen. For each set of FAPs, the proposed algorithm is estimated to obtain utility by setting the maximum demand of PRBs i.e., D from 10 to 20. The utility is calculated as an average value. Comparison

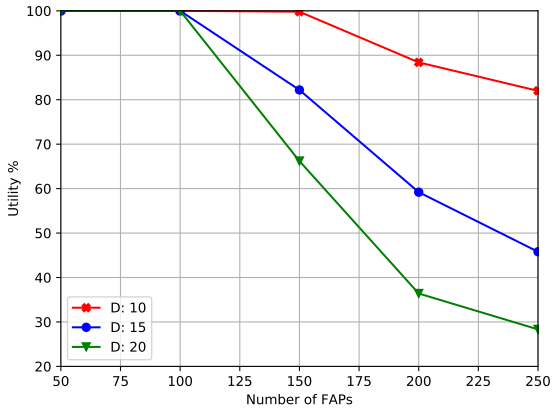

Figure 12: Utility vs. number of FAPs.

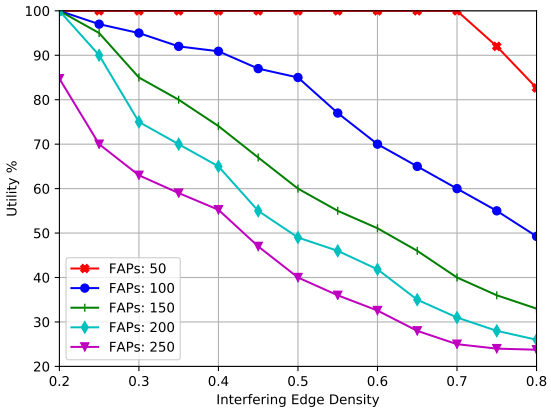

Figure 13: Utility vs. interference link densities.

of utility with respect to FAP is shown in Fig.12. From the graph, we can see that, for a specific demand of resources when there is an increase in total PRB demand, there is a gradual decrease in utility with a sudden fluctuation as number of FAPs approach to 250. This is because as the number of IoT devices increase, there is a throttle for resources in the network leading to a performance decline of the network and hence lower utility value. Additionally, as the demand for resources increase for the same number of FAPs, utility of network resources decrease due to scarcity of resources to meet a higher demand.

B) Impact of Interference Link Density on Utility : Fig. 13 demonstrates the graph for utility v/s interference of link density. Link density is defined as a ratio between actual edges in interference graph to maximum possible edges. This is performed to study impact of network interference on utility of resources among network devices. Utility is different for different link density values.

From the result, we can observe that as link density increases, resource utility starts decreasing gradually. This is because for a given set of FAPs when there is a higher link density there is stronger interference in wireless link that lead to lower resources re-usability. This causes decline in network resource utility of devices.

C) Impact of Number of Allocated Resources on Utility: In Fig. 14, we compare the utilities along with an increment of the number of allocated PRBs. Each iteration of resource requests is independent of the prior iterations. For a given set of IoT devices, as the number of allocated resources in the network increase, there is a sharp increase in the utility as now more resources can be assigned to each IoT device. If the number of allocated resources are small, the devices will not be serviced as per their demand due to interference constraint and hence, smaller utility. However, if the total resources in the network are sufficient, the effective resources allocated to each device is higher, which results in higher utility for the network. Additionally, it is evident from the graph that as the number of IoT devices increase the utility will

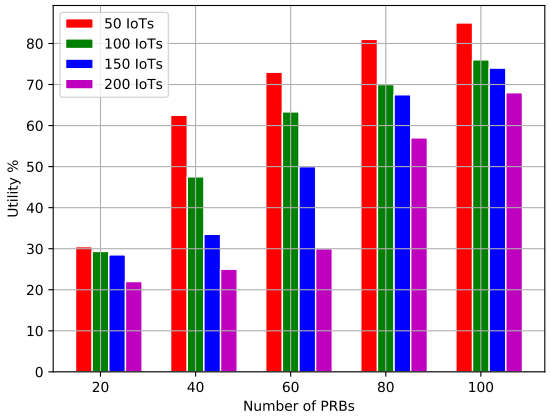

Figure 14: Utility vs. number of PRBs.

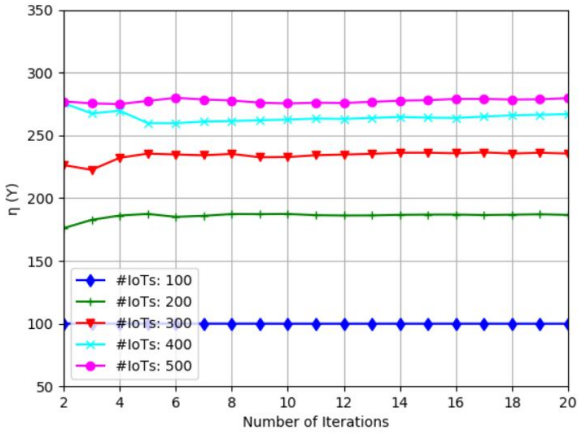

Figure 15: Task throughput vs. number of iterations.

decrease for the given amount of resources. This is because for a specific amount of allocated resources, as the number of devices in the network increase, more subgroups of non-interfering devices will be formed that can lead to a division of resources into more number of portions. Due to this, some devices may fall short of meeting their minimum resource requirement which causes a drop in the utility of the network resources. The results of simulation analysis demonstrate a favourable impact on maximizing task throughput and Network Utility with the proposed algorithms and network parameters under the predetermined constraint of available PRBs.

D) Impact of Number of Iterations on Throughput: We demonstrated the relation between the average number of iterations and throughput in Fig. 15. From the outcome, we conclude that after a fixed number of iterations, throughput becomes constant, and the number of iterations does not add any contributions in overall throughput at the fixed number of IoT devices, FAPs and PRBs. The reason behind this finding is that, in each iteration, several intermediate states are maintained while allocating the resources to each FAP and then to IoTs as described in the above Figs. 3 and 4. Specifically, when the achieved throughput is low due to limited available PRBs, the remaining IoTs can only be covered if allocation to those devices does not create interference in the network, and this phenomenon remains the same across the different iterations in our proposed algorithm.

9.3. Latency Evaluation

In this sub-section, we present a prototype of the proposed architecture to compare the performance of the proposed scheme. Further, we compare the total latency of the developed prototype with the obtained results. We assumed that two PRBs are dedicated to each device in order to transmit the data to respective FAP.

9.3.1. Prototype Specification

To create a prototype of the proposed model, we use a laptop, an android mobile phone, and one raspberry pi as IoT devices sending data over Wi-Fi to another raspberry pi which is working as an FAP in our model.

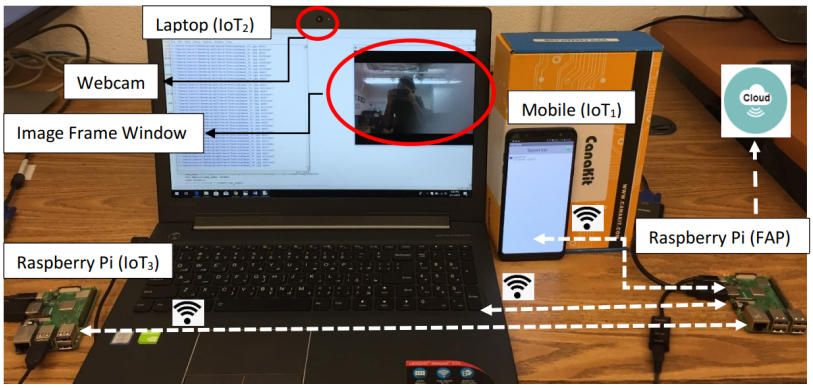

Figure 16: Prototype setup with mobile, laptop, raspberry pi and cloud with their communication medium.

The specification of each device used in the prototype model is shown in Table 3. The camera enabled laptop takes photos in every interval of 1 ms and forwards the images to the central FAP. Another raspberry pi working as IoT device continuously sends stored images to the raspberry pi working as FAP. Additionally, the android mobile phone sends some continuous text messages through a client application to the FAP raspberry pi over the Wi-Fi network. On the FAP side, upon receiving the data from all three IoT devices, the raspberry pi FAP performs local data processing and once the execution is completed it sends response messages to respective IoT devices. We estimate the proposed scheme on different size of data varying from 4 Mb to 209.92 Mb by assuming the task deadline as 1 minute. With the increase of data size, the total time consumed to execute the task also increases. As from Fig. 17 we can observe that the total consumed latency in numerical result and prototype model is almost the same. However, with the increase in data size, the total latency consumed by the prototype model is increasing. The main reason for this difference is that, laptop and smart-phone used as IoT devices are not dedicated devices and these devices are running several other applications in parallel while in use as an IoT device in the prototype model.

9.4. Joint PRB and Latency Evaluation

It is worth mentioning that the entire process of communication between IoT devices (i) and FAPs (k) consists of three stages: (i) IoT devices requesting the BS/FAPs to perform a given task using the up-link channel, (ii) FAPs executing the assigned tasks, and (iii) FAPs reporting their task results back to the respective IoT device using the down-link channel. We assume that the BS leases out the same set of PRBs to the pair for both request and response channels. Considering these stages, we evaluate the total service

Table 3: Hardware specification of prototype

Devices	Specification
Raspberry Pi	**Model:** Raspberry Pi 3 B+, **SOC:** Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC. **CPU:** 1.4GHz 64-bit quad-core, ARM Cortex-A53 CPU, **RAM:** 1GB SDRAM, **WiFi:** Dual-band 802.11ac wireless LAN, (2.4GHz and 5GHz) Bluetooth 4.2. **Power:** 5V/2.5A DC power input.
Laptop	**Model:** Lenovo Ideapad 510. **Processor:** Core i5 7th Gen. **CPU:** 2.5 Ghz. **RAM:** 2GB SDRAM, **WiFi:**802.11 a/b/g/n/ac, Bluetooth 4.2 **Web-Cam:** Yes, recording at 720p HD, **Battery:** Cell Li-Ion.
Mobile	**Model:** Samsung Galaxy A6 +. **OS:** Android 8.0 (Oreo). **CPU:** Octa-Core 1.8GHz Cortex A53. **RAM:** 4GB, **WiFi:**8802.11 a/b/g/n, WiFi Direct, hotspot and Bluetooth 4.2 **Network technology:** GSM/HSPA/LTE.

latency into two parts *a)* Transmission latency *b)* Processing latency. This total communication delay is represented as follows:

$$\tau_{k,i} = \tau_{k,i}^{\text{send}} + \tau_{k,i}^{\text{proc}} + \tau_{k,i}^{\text{recv}}$$

The transmission latency comprises of the up-link ($\tau_{k,i}^{\text{send}}$) and the down-link ($\tau_{k,i}^{\text{recv}}$) latency between the IoT devices and the corresponding FAPs. The up-link transmission time corresponds to the time taken by a mobile IoT user to offload the computational task to it's respective FAP via a wireless channel with the bandwidth (Δf). It is computed as the ratio of data size (i.e. total demand in the network) with respect to the transmission data rate for offloading. It is important to note that in a mobile cellular communication the

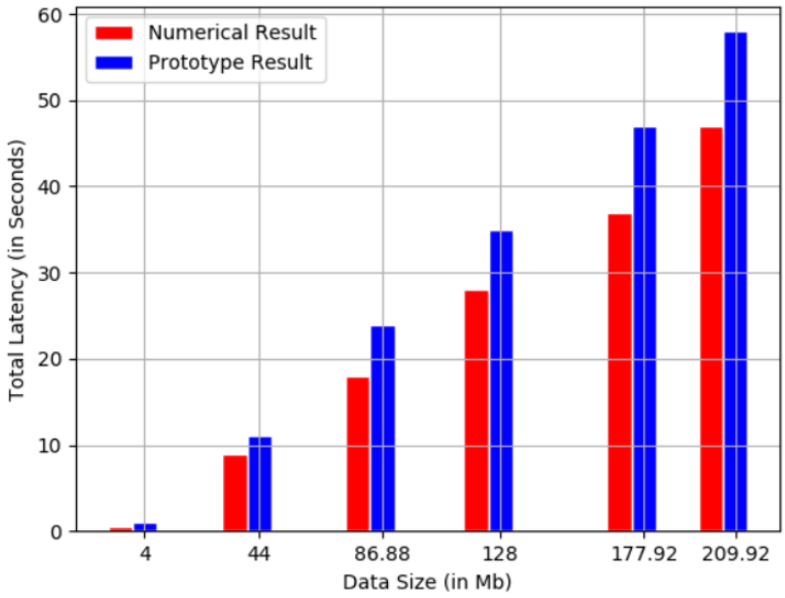

Figure 17: Comparison of total delay and data size on developed prototype and numerical result.

data received after processing from FAPs is very small in contrast to the initial data. Thus, we consider the down-link latency($\tau_{k,i}^{\text{recv}}$) to be relatively trivial and assume it to be a random value such that $\tau_{k,i}^{\text{recv}} \in [0, 1]$. On the other hand, the processing latency accounts for the execution time across the FAP nodes and depends on the CPU processing rate. In order to compute the total service latency the network operational parameters for the data rate, the CPU processing rate, and CPU processing cycles per bit is based on the general OFDMA model [14], [20], and [21]. Fig. 18 exhibits the latency comparison for the proposed scheme with SPA [10] and JELO algorithms [12]. We draw a random topology of 20-100 number of IoT devices randomly distributed in a network of 5 FAPs with an interference range of 50 m. We assumed that the computational tasks generated from each IoT device sends 1 Mb of data to get executed at each corresponding FAP. The data rate for communication between IoT devices and FAP nodes is set to 1008 bits carried by every PRB pair per unit of time. The CPU computational rate for processing at each FAP is set to a value of 1.4 GHz. For the work [12], we keep the balance coefficient $\alpha = 0$ in order to compare the latency with our proposed scheme. We can conclude from Fig. 18, with the increase in the number of PRBs in the network total latency, is minimized. The reason from this observation is that when the more number of PRBs are allocated for the same task, the total achievable data rate goes high and propagation latency becomes low, respectively. As shown in Fig. 18, amongst the three methods, the SPA algorithm takes the highest average latency than the proposed scheme. This is because in the SPA algorithm each IoT device is assigned only a share of the available CPU and radio resource. It assumes that only one resource is assigned to each IoT user such that multiple devices share the same channel within it's maximum capacity leading

to a downgrade for the transmission rate and hence higher latency. Similarly the JELO scheme addresses the task assignment problem on the FAP nodes and simply allocates a set of resources to IoTs based on the volume of arriving tasks at FAPs causing higher transmission latency in the network. Our proposed scheme gives the lowest latency compared to the existing algorithms. The reason is that unlike the existing works our proposed scheme aims at maximizing the re-usability of PRBs while avoiding interference among FAPs and this phenomenon improves the achievable data rate between FAP and IoT device. Consequently, our proposed scheme results in a lower latency in the network. Thus, the proposed scheme outperforms both the SPA and JELO algorithms with a percentage improvement of 27% and 21% respectively.

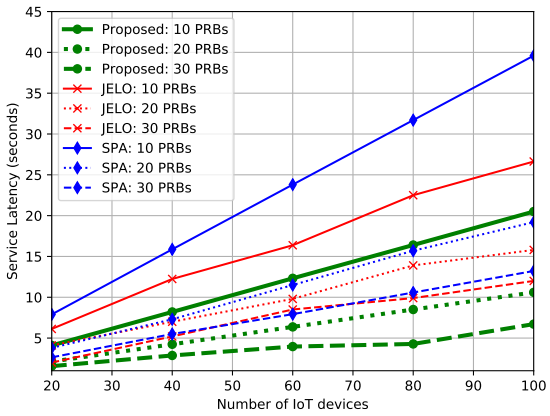

Figure 18: Service latency vs. IoTs.

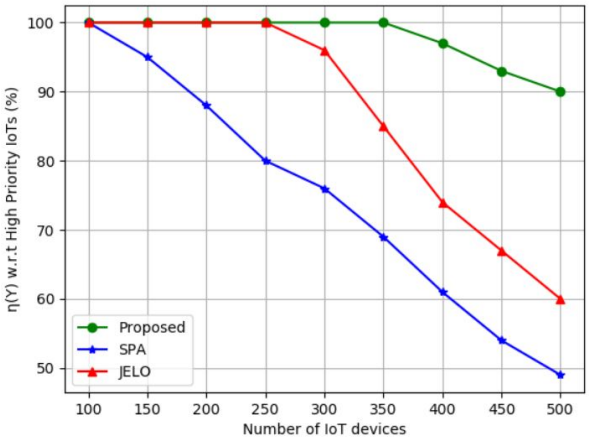

Figure 19: Throughput vs. IoTs.

9.5. Analysis of Throughput

To compare the throughput with JELO and SPA approaches, we simulated an environment of 50 FAPs, 100 PRBs, and IoT devices in a range of [100,500] with an evenly distributed demand for high and low priority tasks. Fig. 19 represents the performance of each allocation scheme in terms of task throughput as a percentage with respect to high priority IoT demand in the network. From Fig. 19, we can observe that at 100 IoTs, all three proposed schemes can satisfy the demand for high priority devices fully with 100% task throughput. But the performance of all three curves is decreasing with the increase in number of IoT devices or demand in the network. In SPA, each IoT-FAP pair selects a PRB with the best sum task throughput. Even though the greedy behaviour of the algorithm helps it to achieve a higher task throughput, it starves the high priority IoT devices. As the IoT devices increase to 500, the task throughput curve converges to almost 55%. Alternatively, JELO assumes that an equal share of PRBs (corresponding to each task) are assigned to the IoT-FAP pair while considering the task priorities. This reduces the burden of overdue un-serviced demand from high priority IoT devices. Hence, the task throughput decreases at a slower rate

in contrast to the SPA scheme. In JELO, as the IoT count reaches to more than 350, less than 80% of high priority IoT devices are being serviced and when the IoT count is set to 500 the JELO curve falls to 60%. On the other hand, it is evident that the proposed scheme outperforms the other two allocation schemes reaching to almost 90% task throughput for satisfying high priority IoT devices with the total IoT count set to 500. Thus, the proposed scheme considers maximizing resource utilization in the network while taking care of individual IoT demand and priorities effectively.

9.6. Real World Data Analysis

We executed our proposed algorithm on a real world dataset of base stations and the end users within the Metropolitan Melbourne in Australia. The dataset is obtained from the work [23]. The geographical location of base stations is considered as the location of Fog server because Fog servers are usually deployed at base stations [24]; and end users are the IoT devices in our framework. The coverage area of FAPs is set to 500 meters to derive the interference graph. Since the distribution of IoT devices is highly skewed, we consider a subset of these IoT devices that are in close proximity to the FAP nodes in the network. Consequently, the total number of FAP nodes and IoT devices in the network topology are 125 and 94. Fig. 20 illustrates this distribution of IoT and FAPs in terms of their geographic location coordinates (i.e. latitude and longitude) which is mapped into the network topology.

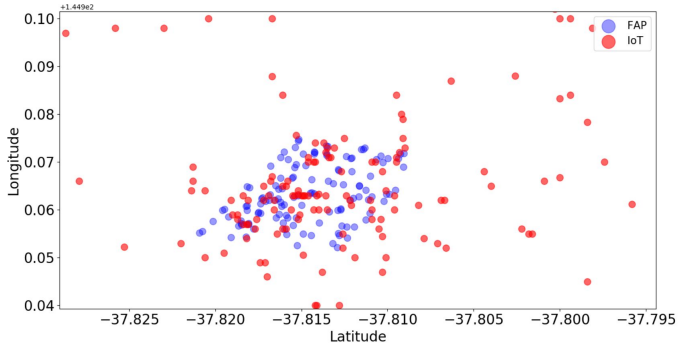

Figure 20: Distribution of FAPs and IoTs.

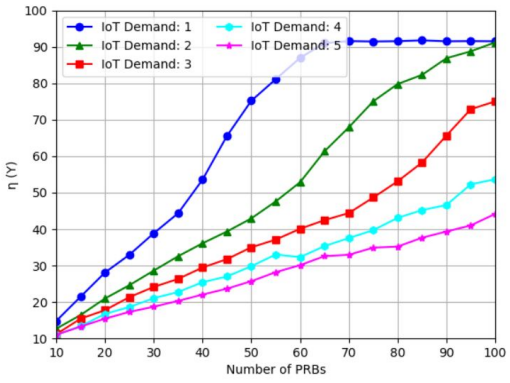

Figure 21: Variation of $\eta(Y)$ and number of PRBs.

As observed in Fig. 21, for a given demand in the range of [1,5], the task throughput, $\eta(Y)$, is positively co-related with the number of available PRBs in the network. Setting the maximum IoT demand to 1, as the number of PRBs increases to 100, the value of $\eta(Y)$ increases proportionally and converges at PRB count of 70; in this case the task throughput is 94, thereby servicing the full demand of all the IoT devices in the network. When the maximum demand is set to 2 and the the PRB count is 100, the task throughput of 94 is still achieved due to the re-usability of resources among non-interfering FAPs. This yields a high network

resource utilization in the network. As the maximum IoT demand increases from 2 to 5, the $\eta(Y)$ decreases gradually. At a maximum demand set to 5 with 100 PRBs, the proposed algorithm is able to serve about 50% of the IoT devices ($\eta(Y) = 45$) due to increased throttle for the available resources. These results are in accordance with theoretical and simulation results, thus validating our method in a real-world scenario.

10. Conclusion

In this work, we proposed a novel framework for IoT and fog enabled 5G networks where the BS identifies appropriate pairs of IoT devices and FAPs and allocate necessary resources to maximize the total number of tasks served. We proposed a graph-coloring based algorithm to solve it in a computationally tractable manner. Through numerical result and prototype model, we have demonstrated the effect of different parameters over the utility and latency of IoT devices in different environments. From the simulation and experimental results, we found that the proposed algorithm overcomes the caveats in existing approaches in terms of both maximizing task throughput and achieving low latency with a percentage improvement of more than 20% on an average. Moreover, we have also considered resource assignment based on IoT tasks priorities which existing methods have not taken into account. In the future, the proposed scheme can be applied to specific applications and dynamic network model via considering specific applications by modifying the set of constraints accordingly. Additionally, since our current discussions focus only on computational issues that arise within the single-iteration problem setting, we also intend to study the case of starvation of resources across multiple iterations which is a significant problem.

Acknowledgments: The authors are grateful to the anonymous reviewers and guest editors for insightful comments and constructive suggestions that helped improve the quality of the manuscript significantly. This work was partially supported by NSF grants CCF-1725755, CNS-1545050, and CNS-2008878.

References

References

[1] N. Hassan, K.-L. A. Yau, C. Wu, Edge computing in 5G: A review, IEEE Access 7 (2019) 127276–127289.

[2] O. Givehchi, K. Landsdorf, P. Simoens, A. W. Colombo, Interoperability for industrial cyber-physical systems: An approach for legacy systems, IEEE Tran. on Ind. Inf. 13 (6) (2017) 3370–3378.

[3] A. Pratap, R. Singhal, R. Misra, S. K. Das, Distributed Randomized k-Clustering Based PCID Assignment for Ultra-Dense Femtocellular Networks, IEEE Transactions on Parallel and Distributed Systems 29 (6) (2018) 1247–1260.

[4] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog computing and its role in the internet of things, in: Proceedings of the first edition of the MCC workshop on Mobile cloud computing, ACM, 2012, pp. 13–16.

[5] R. Atat, L. Liu, H. Chen, J. Wu, H. Li, Y. Yi, Enabling cyber-physical communication in 5G cellular networks: challenges, spatial spectrum sensing, and cyber-security, IET Cyber-Physical Systems: Theory & Applications 2 (1) (2017) 49–54.

[6] A. Pratap, F. Concone, V. S. S. Nadendla, S. K. Das, Three-Dimensional Matching based Resource Provisioning for the Design of Low-Latency Heterogeneous IoT Networks, in: Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2019, pp. 79–86.

[7] F. Samie, V. Tsoutsouras, S. Xydis, L. Bauer, D. Soudris, J. Henkel, Distributed QoS management for Internet of Things under resource constraints, in: Proceedings of IEEE/ACM/IFIP Int. Conf. on Hardware/Software Codesign and System Synthesis, ACM, 2016, p. 9.

[8] S. F. Abedin, M. G. R. Alam, N. H. Tran, C. S. Hong, A Fog based system model for cooperative IoT node pairing using matching theory, in: APNOMS, 2015, IEEE, 2015, pp. 309–314.

[9] H. Zhang, Y. Xiao, S. Bu, D. Niyato, F. R. Yu, Z. Han, Computing resource allocation in three-tier iot fog networks: A joint optimization approach combining stackelberg game and matching, IEEE Internet Things J. 4 (5) (2017) 1204–1215.

[10] Y. Gu, Z. Chang, M. Pan, L. Song, Z. Han, Joint radio and computational resource allocation in iot fog computing, IEEE Transactions on Vehicular Technology 67 (8) (2018) 7475–7484. doi:10.1109/TVT.2018.2820838.

[11] M. Aazam, K. A. Harras, S. Zeadally, Fog computing for 5G tactile industrial internet of things: Qoe-aware resource allocation model, IEEE Transactions on Industrial Informatics 15 (5) (2019) 3085–3092.

[12] D.-N. Vu, etal., Joint energy and latency optimization for upstream IoT offloading services in fog radio access networks, Tran. on Emerging Tel. Tech. (2018) 1–14.

[13] M. Peng, etal., Fog-computing-based radio access networks: issues and challenges, Ieee Network 30 (4) (2016) 46–53.

[14] A. Pratap, R. Misra, S. K. Das, Resource Allocation to Maximize Fairness and Minimize Interference for Maximum Spectrum Reuse in 5G Cellular Networks, in: IEEE WoWMoM, IEEE, 2018, pp. 1–9.

[15] A. Pratap, R. Misra, S. K. Das, Maximizing Fairness for Resource Allocation in Heterogeneous 5G Networks, IEEE Transactions on Mobile Computing (2019) 1–1doi:10.1109/TMC.2019.2948877.

[16] A. Pratap, S. Singh, S. Satapathy, S. K. Das, Maximizing Joint Data Rate and Resource Efficiency in D2D-IoT Enabled Multi-Tier Networks, in: 2019 IEEE 44th Conference on Local Computer Networks (LCN), IEEE, 2019, pp. 177–184.

[17] Y.-S. Liang, W.-H. Chung, G.-K. Ni, Y. Chen, H. Zhang, S.-Y. Kuo, Resource allocation with interference avoidance in OFDMA femtocell networks, IEEE Transactions on Vehicular Technology 61 (5) (2012) 2243–2255.

[18] D. Bertsimas, J. N. Tsitsiklis, Introduction to linear optimization, Vol. 6, Athena Scientific Belmont, MA, 1997.

[19] D. S. Johnson, M. R. Garey, Computers and Intractability: A Guide to the Theory of NP-completeness, WH Freeman and Company, New York, 1979.

[20] A. Hatoum, R. Langar, N. Aitsaadi, R. Boutaba, G. Pujolle, Cluster-based resource management in OFDMA femtocell networks with QoS guarantees, IEEE TVT 63 (5) (2014) 2378–2391.

[21] ETSI, LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical channels and modulation (3GPP TS 36.211 version 14.3.0 Release 14), 2017-08.

[22] D. J. Welsh, M. B. Powell, An upper bound for the chromatic number of a graph and its application to timetabling problems, The Computer Journal 10 (1) (1967) 85–86.

[23] Q. He, G. Cui, X. Zhang, F. Chen, S. Deng, H. Jin, Y. Li, Y. Yang, A game-theoretical approach for user allocation in edge computing environment, IEEE Transactions on Parallel and Distributed Systems 31 (3) (2020) 515–529.

[24] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, V. Young, Mobile edge computing—A key technology towards 5G, ETSI white paper 11 (11) (2015) 1–16.