DEFT: Decentralized Multiuser Computation Offloading in a Fog-Enabled IoV Environment

Pallav Kumar Deb, *Student Member, IEEE,* Chandana Roy, *Student Member, IEEE,*
Arijit Roy, *Student Member, IEEE,* Sudip Misra, *Senior Member, IEEE*

***Abstract*—In this paper, we propose a mechanism –DEFT– to facilitate mobile Internet of Vehicles (IoV) user entities (UEs) with unit tasks to cooperate among one another and offload their tasks to nearby fog nodes (FNs)/roadside units (RSUs) in a decentralized and fair manner. While existing literature depends on offloading schemes based on centralized systems, DEFT provides near real-time computation offloading in a fog-enabled IoV using a decentralized two-level game-theoretic approach. Further, this approach allows the UEs to make their own decisions in a dynamic environment. In the first level, FNs play an *Oligopoly* game for determining the price of their services based on their computation capability and then broadcast the price to the UEs. Further, based on the price transmitted by the FNs, UEs play a non-cooperative *Stackelberg* game to map the offloading among the user and fog layer. We also ensure that no UE consumes all of the resources in an FN. Additionally, we prove our objective function's convexity and show that the proposed game always attains the Nash Equilibrium. Through extensive real-world emulations, we observe that DEFT minimizes the latency and power consumption, and improves throughput compared to the existing schemes. In the presence of 40, 80, and 120 UEs, the overhead for computation in the fog layer is minimized by $70-75\%$ compared to the local computation overhead.**

***Keywords*—Internet of Things, Fog Computing, Decentralized Computation Offloading, Dynamic Pricing, Game Theory.**

I. Introduction

Vehicles in an Internet of Vehicles (IoV) environment comprises of multiple sensors such as accelerometers, cameras, infrared sensors, and others [1], [2]. These sensors collect data in different formats, which require complex operations for making inferences. Further, complex processing of the collected data is necessary for the IoV user entities (UEs) to perform local computation. Additionally, these tasks are required to make real-time decisions for the actuators, control systems, and other machinery present in the vehicles. To save local resources and energy in an IoV environment, offloading the tasks to external platforms such as cloud and fog computing paradigms prove beneficial [3].

In this paper, we propose a decentralized computation offloading scheme DEFT, by adopting a game-theoretic approach for mobile IoV UEs to efficiently and fairly select fog nodes (FNs) within their proximity. Fig. 1 depicts an overview of our proposed solution. We consider that the FNs and the UEs communicate using radio access network (RAN) technologies. Initially, the FNs interact with one another to determine a uniform price for their services, based on their computational capability, storage, and waiting time. To achieve this, we formulate an oligopoly game. Further, after the price is determined, we formulate a Stackelberg game to select the appropriate FN, where the UEs act as players. We envision DEFT as a scheme for the UEs to independently select the FNs without any centralized entity, removing the possibility of single-point failures.

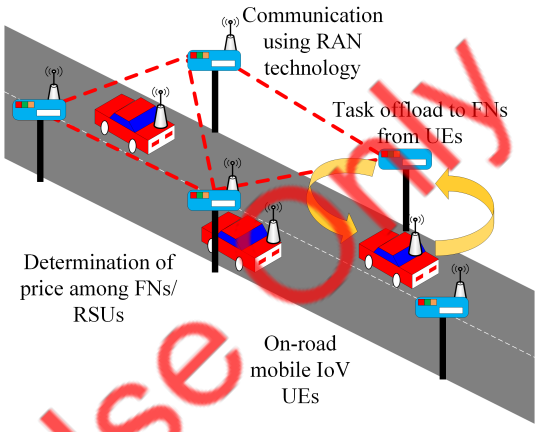


Fig. 1: Overview of the proposed scheme DEFT for computation offloading in mobile IoV environments

Computation Offloading [4], [5], is a well-explored possibility, and the current literature offers multiple solutions towards offloading the load of performing complex operations to external platforms, such as the cloud [6], [7]. However, with the production of sizeable time-sensitive data by the devices in an IoV network, we may lose the opportunity to act efficiently and make uninformed decisions. These issues mandate the need for device-to-device communication-enabled solution techniques. We select the fog computing paradigm to minimize the delay incurred [8]. Typically, the fog nodes/roadside units (RSUs) (in case of IoV) consist of devices such as *switches*, *routers*, *gateways*, *bridges*, and *hubs*, which are present at the edge of the network. These devices are usually resource-constrained compared to the cloud servers and include a varied range of devices. Such device configurations motivate us to design solutions which ensure FNs' availability based on the user's task and also consider the distance between them. The selection of the FNs based on their configuration and distance reduces processing and propagation delays. Existing solutions rely mostly on centralized systems, which are vulnerable to single-point failures, bottlenecks, and malicious attacks. We reduce the possibility of these threats by adopting a decentralized approach. These facts act as a motivation to design our fog-based decentralized architecture and mechanism while accounting for the varying device configurations. Although the decentralized solutions add overheads in the network for passing control messages, it

P. K. Deb is with the Department of Computer Science and Engineering, C. Roy is with the Department of Industrial & Systems Engineering, A. Roy is with the Advanced Technology Development Centre, and S. Misra is with the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India, Email: {pallv.deb, chandanaroy, arijitroy, sudipm}@iitkgp.ac.in

allows us to realize the autonomy in the devices, particularly for the FNs, to tailor their configurations and services according to the UEs' requests. To unanimously determine the price of services from the FNs and decide which UE offloads to a particular FN, we propose a real-time decentralized game-theoretic scheme.

A. Why Game Theory?

Compared to the conventional solution methods, game-theoretic solutions allow its participants (users and fog nodes) to *interact* with one another and identify the possible set of *strategies*. Game theory also has the feature of maintaining clarity among the outcomes and follows a discrete nature. Such traits give us the *flexibility* of assigning custom rewards. Further, these salient features enable the devices to be aware of the system's rules of engagement, allowing them to make smart decisions accordingly. Further, in the presence of alternative strategies, the participants select the one which renders the highest benefit. In this work, we consider a set of FNs offering computational services to the UEs with both parties acting independently. Moreover, as we designed separate models for determining the price of the services, based on the available resources in the FNs and its consumption by the UEs, game-theoretic solutions are a perfect fit.

In this paper, we primarily aim to address the following issues: (a) How the FNs set the price for the computation of the offloaded data? and (b) Based on the price transmitted by FNs, which FN is selected by the UEs. To enable fairness among the UEs, we introduce penalties such that they do not consume all the available resources at an FN. Such penalties help in ensuring that resources are available for the other UEs requesting for computational services. The specific *contributions* of this work are as follows:

- We propose a decentralized, two-tier scheme, named as DEFT, for mobile IoV UEs to connect and offload their tasks to the FNs with ease. The UEs cooperate among them and offload their tasks to the nearest FNs.
- We formulate an oligopoly game to map the interactions among the FNs. The FNs then determine the price for their services, depending upon their residual storage space, computation capability, and time for which the tasks wait in the queue.
- We formulate a cooperative Stackelberg game among the UEs to select the appropriate FNs. These UEs are capable of dynamically switching among the FNs whenever there is scope for increasing their utility.
- Extensive simulation of our proposed scheme depicts that DEFT outperforms in terms of latency, throughput, and power consumption, compared to the existing schemes [8], [9].

The remaining of the paper is organized as: we briefly provide insights towards the existing related researches on computation offloading in Section II followed by the problem description in Section III. We then present our observations while evaluating DEFT in Section IV and finally draw conclusions in Section V.

II. Related Work

In this section, we discuss some related research works in the field of computation offloading in fog and cloud. As the UEs are resource-constrained and sophisticated applications run on them, the performance and lifetime of the UEs degrade. Considering these facts, Lin *et al.* [10] proposed an offloading framework, named as Ternary Decision Maker (TDM) to minimize the response time and energy consumed. Additionally, the authors performed real-world applications to evaluate the performance of TDM. Similarly, Samanta *et al.* [11] considered delay-tolerant and resource-constrained mobile devices. They proposed an adaptive offloading scheme to maximize the revenue of the mobile edge computing (MEC) scenario. Further, Chen *et al.* [5] proposed a centralized computation offloading problem in the presence of multiple users for a mobile edge cloud computing environment. The authors proved that the problem is NP-hard and applied a game-theoretic approach for achieving efficient computation offloading. On the other hand, from the consumers' perspective, there exists uncertainty in cloud services, due to lack of transparency in the type of services, operational conditions, and quality of service given by multiple service providers. Emeakaroha *et al.* [12] designed a system to improve the trust of consumers and their trustworthiness in cloud services. They evaluated the consumers' trust in cloud services, depending on the use cases.

Meskar *et al.* [13] proposed a cloud-based computation offloading approach to improve the energy consumption of mobile users. Considering real-time constraints such as job execution deadlines, user-specific channel bit rates, and competition across shared channels, the authors modeled the problem scenario as a competitive game. Similarly, to improve the service quality and performance of mobile edge cloud computing, Tao *et al.* [14] proposed an energy minimizing computation offloading scheme. Further, the authors also considered the bandwidth capacity at each time slot in their proposed offloading approach. Zhou *et al.* [15] transformed the computation offloading problem into a matching problem by introducing contracts to maximize the expected utility. However, the matching problem increases the time complexity of arriving at an optimal solution. Keeping in mind the upcoming AI technologies, Zhao *et al.* [16] proposed an offloading scheme for fog RANs.

Synthesis: The existing literature provides solution to several problems related to offloading in both cloud and fog computing, such as response time minimization [10], energy consumption [11], and customer trust [12]. However, these solutions assume uniformity for the nature of devices. The proposed decentralized computation offloading scheme, DEFT, provides solutions to task offloading in the presence of *heterogeneous* types of devices (both FNs and UEs). In this work, we propose a two-level *game-theoretic*, *decentralized* scheme to offload the tasks to the FNs dynamically. The proposed scheme provides a solution to *pricing* the services of the FNs. These prices reflect the resources available at the FNs. We also ensure that no UE selfishly consumes the resources in its entirety by imposing *penalties*. Such penalties help in maintaining *fairness* among the devices (both the FNs and UEs).

III. Problem Description

In this section, we describe our problem scenario and present the mathematical formulation for the two-tier game-theoretic approach for offloading the tasks to the fog layer.

A. Problem Scenario

We consider an Intelligent Transportation System (ITS) scenario, where vehicle-to-vehicle (V2V) and vehicle-to-

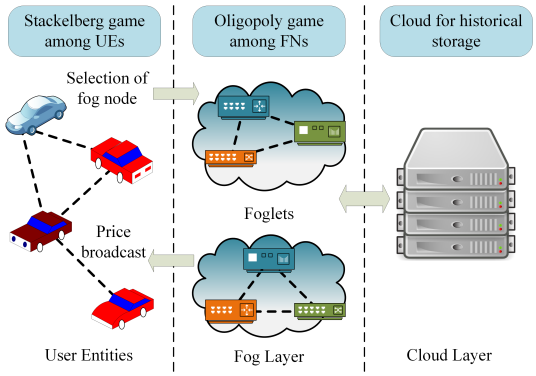


Fig. 2: System Architecture

infrastructure (V2I) communication is possible. Fig. 2 shows a foglet consisting of FNs connected in the form of a mesh network. In IoV scenarios, foglets are a set of RSUs within the same network or the same vicinity. It may be noted that we refrain from the creation of the foglets to maintain simplicity. Further, with the mobility of the UEs, they either enter or exit from these foglets. We assume that the UEs are in a quasi-static state within the network, such that their tasks get executed during the time they are present in the foglet. This assumption allows us to return the results through the same FN. The type of devices (both UEs and FNs) present vary from one another in terms of their configurations, which introduces heterogeneity in the problem scenario. Additionally, as both FNs and UEs communicate through RAN technologies, we assume that the UEs seamlessly connect to any of the FNs upon requirement.

Assumptions: The list of assumptions in this work are as follows:

- We consider that the fog nodes do not distribute the task execution with the cloud platform. The cloud is only necessary for storage purposes. We plan to extend this work by enabling task redistribution among the fog nodes to facilitate task parallelism.
- Inspired from the works of Lei *et al.* [17], we assume that the fog nodes are connected through a strong backhaul network. This connection among the fog nodes may follow the same architecture as the works of Fan and Ansari [18].

B. Problem Formulation

We consider the presence of a set of UEs and FNs in our IoV scenario. They vary with respect to their computational capabilities, battery capacity, and nature of tasks. Let $U = \{u_1, u_2, u_3, ..., u_{\mathcal{N}}\}$ be the set of UEs present in the network. These UEs have tasks/jobs (J_i) corresponding to each UE, u_i for execution. We represent these tasks using a two-tuple format as $J_i = \langle I_i, D_i \rangle$, where I_i is the size of the input data (such as program codes and input parameters) related to the task, J_i and D_i denote the number of CPU cycles required to execute the i^{th} task, J_i. The UEs need to assign their tasks to the FNs in an optimal manner to cope with the time-sensitive constraint. In this work, we propose a two-tier game-theoretic approach to offload the FNs.

Let F_i^{dev} denote the computation capability of each UE, u_i, in CPU cycles per second. The time, T_i^{dev} required by them to perform the tasks locally is represented as, $T_i^{dev} = \lceil D_i / F_i^{dev} \rceil$. Further, β_i represent the energy consumption per unit CPU cycle. The amount of energy consumed by the i^{th} device to perform the task is mathematically represented as, $E_i^{dev} = \beta_i D_i$. Suppose T_{max}^{dev} and E_{max}^{dev} are the maximum time required for the completion of a task and the maximum energy consumed when all the resources of the UE are in use. We compute the overall overhead of the device for performing the task locally as a sum of the normalized time and energy. Inspired from the works in [19], we add weights $w_t^{dev_i}$ and $w_e^{dev_i}$ in our formulations. Therefore, the overhead (Z_i^{dev}) for local processing is:

$$Z_i^{dev} = \left\lceil w_t^{dev_i} \frac{T_i^{dev}}{T_{max}^{dev}} + w_e^{dev_i} \frac{E_i^{dev}}{E_{max}^{dev}} \right\rceil \quad (1)$$

where $(w_t^{dev_i}, w_e^{dev_i}) \in (0, 1)$ and $w_t^{dev_i} + w_e^{dev_i} = 1$. These weights represent the preference of the UEs. Based on their current state and requirement, these weights are determined. For instance, when the UE has a low battery, it sets a higher $w_e^{dev_i}$ to save more energy. On the other hand, when the UE runs some time-sensitive application (video streaming), it sets a higher $w_t^{dev_i}$ value to save time. We determine these weights based on different power saving modes. For instance, the value of $w_e^{dev_i}$ is set to 1, when the battery is deficient and activate (extreme) power-saving mode. Other values of $w_e^{dev_i}$ may be varied proportionately to the available battery. Since $w_e^{dev_i}$ is related to $w_t^{dev_i}$, we set $w_e^{dev_i} = 0$, in case of time-sensitive applications and enter extreme performance mode (only when the available battery permits). Such methods allow the UEs to set weights dynamically and switch according to the operation modes. In summary, these weights enable the UEs to analyze their status.

Similarly, we compute the overhead for assigning the tasks to the FNs. Let $G = \{g_1, g_2, ..., g_{\mathcal{M}}\}$ be the set of FNs. These devices have different computational capability (F_j^{fog}), storage space (S_j^{fog}), and a queue of pending tasks (t_j^{busy}). We assume that each FN broadcasts a beacon periodically, which comprises $\langle F_j^{fog}, S_j^{fog}, t_j^{busy} \rangle$. Thus, any UE present within the proximity of the FN possess these information. Considering reliable communication links present among these devices, and assuming that each packet is R bits in size, the UEs forward the input data parameters of size, I_i to the FNs when the Shannon Capacity formula is satisfied. Mathematically, $r(\Re[t], P_{tx}) = B \log_2 \left(1 + \frac{\Re[t] P_{tx}}{N_o B}\right) \geq R$ where P_{tx} is the transmission power constraint, B represent the bandwidth, and the noise factor is denoted by N. The time required to offload the input parameters by the UE and the energy consumed during this time is $T_j^{off} = \left\lceil \frac{I_i}{r(\Re[t], P_{tx})} \right\rceil$ and $E_j^{off} = \left\lceil \frac{P_{tx} I_i}{r(\Re[t], P_{tx})} \right\rceil$. On successfully receiving the input parameters, the time required to execute the task is $T_j^{fog} = \left\lceil \frac{D_i}{F_j^{fog}} \right\rceil$. Therefore, considering the maximum time (including transmission and computation) and energy required as $T_{max_j}^{fog}$ and $E_{max_j}^{dev}$, the total overhead for the UEs to execute the task by a FN along with the weight parameters is mathematically represented as:

$$Z_j^{fog} = \left\lceil w_t^{dev_i} \frac{(T_j^{off} + T_j^{fog})}{T_{max_j}^{fog}} + w_e^{dev_i} \frac{E_j^{off}}{E_{max}^{dev}} \right\rceil \quad (2)$$

FN selection by UEs: Due to the limited transmission range of wireless communication technologies (WiFi in this

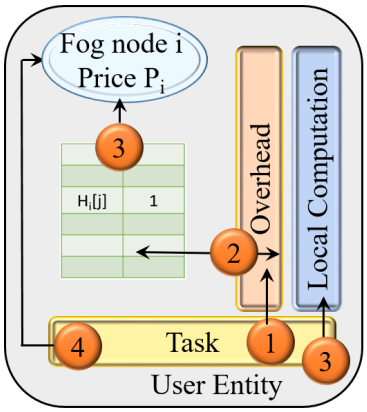


Fig. 3: Service Architecture for a UE

case), the UEs are only aware of the FNs present within their proximity. The UEs store these FNs' information in a hash table on receiving the beacon signals. The UEs use the configurations of these proximal FNs for finding the hash functions to use as identifiers. The hash table also has an added column against each device, which holds binary values to represent selection for offloading. We illustrate using an example. We consider a hash table, H. The FN with the least overhead is assigned the value as 1, and the value as 0 for other FNs. The FN with the value 1 is selected, and the UE offloads their task. Mathematically:

$$H_i[j] = \begin{cases} 0, & \text{if task is not offloaded} \\ 1, & \text{if task is offloaded} \end{cases} \tag{3}$$

where, j in $H_i[j]$ is the identifier for the j^{th} FN present within the proximity of the i^{th} UE. Since the result that returns to the UE is much smaller in size than the input parameters, we assume that the response time is negligible.

Service Architecture: In the proposed decentralized computation offloading scheme, each of the UEs possesses a service architecture. Fig. 3 illustrates the step-by-step representation of the service architecture of UE. Step 1 computes the overhead for both local and remote computation and decides whether to offload the collected data or process it locally. On the other hand, during the local computation, the UE transfers the task to its local processor for execution. Further, in Step 2, if the decision is to offload the data, the UE selects an FN based on the charged price. Upon selection of the FN, the UE updates the hash table H, as mentioned in Section III-B. In Steps 3 and 4, the UE transmits the task to the corresponding FN for execution. Finally, the UE receives the result from the FN after execution.

C. Game Formulation

We propose a cooperative game-theoretic approach to decide when, how, and to whom the UEs offload the tasks. The decisions are a result of strategies from two stages. At the inception level, the FNs play an oligopoly game and decide upon a price for their resources, based on their computation capability, F_g, available storage space, S_g, and the time required by the FN to complete the execution of the tasks existing in the queue, t_{q,g_i}. As a result of the oligopoly game, let $\mathbb{P}_i$ be the price that the FN g_i broadcasts to the UEs. On receiving the price $\mathbb{P}_i$ from each of the FNs within their proximity, the UEs start interacting with the other UEs present within their transmission range. Thereafter, we map the interaction among the UEs as a cooperative Stackelberg game. As an outcome of both the games, the UEs start offloading data in an optimized manner.

Suppose, the per unit computation cost at the i^{th} fog node is C_{g_i} and the computation capability is F_{g_i}, such that $F_{g_i} \leq F_{g_i}^{max}$, where $F_{g_i}^{max}$ is the maximum achievable computation power in CPU cycles per unit time. To characterize each of the FNs, we define the following characteristics:

***Definition* 1.** *Residual Storage ($S_{g_i}^{res}$): The normalized residual storage available at the i^{th} FN is the ratio of the storage space available at time instant, t, coupled with a factor α, such that $0 \leq \alpha \leq 1$, to the maximum available storage. The value of α is the weight determined by the state of the respective FN to show the frequency of storage usage over time. The precise available storage space at any particular time instant is challenging and is statistically analyzed. Mathematically, $S_{g_i}^{res} = \lceil \alpha S_{g_i}^{t} / S_{g_i}^{max} \rceil$, where T is the total number of time-slots. Further, the available storage space and the maximum storage space of the i^{th} FN during the time slot t is represented by $S_{g_i}^{t}$ and $S_{g_i}^{max}$.*

***Definition* 2.** *Normalized Computation Capability ($F_{g_i}^{norm}$): The normalized form of the computation capability of the i^{th} FN is the ratio of the present computation capability, F_{g_i}, and the maximum achievable value, $F_{g_i}^{max}$. Therefore, the normalized computation capability is $F_{g_i}^{norm} = \lceil F_{g_i} / F_{g_i}^{max} \rceil$.*

***Definition* 3.** *Waiting Time ($t_{g_i}^{wait}$): The normalized waiting time is the ratio of the summation of the time required for the execution of each of the tasks in the queue of i^{th} FN and the maximum tolerable time, t_{q,g_i}^{max}. Therefore, the value of $F_{g_i}^{norm}$ reduces to a certain extent. Mathematically, $t_{g_i}^{wait} = \lceil t_{q,g_i} / t_{q,g_i}^{max} \rceil$.*

***Proposition* 1.** *The price that any FN charges to endure for the execution of a task assigned to it by some arbitrary UE is an estimation of the cost function of the residual storage $S_{g_i}^{res}$, computation capability $F_{g_i}^{norm}$, and the waiting time $t_{g_i}^{wait}$. Therefore, the price charged by an FN is mathematically represented as,*

$$\mathbb{C}_{g_i} = C_{g_i} \frac{F_{g_i}^{norm}}{S_{g_i}^{res} t_{g_i}^{wait}} \tag{4}$$

where, C_{g_i} is the per unit computation cost for the i^{th} FN.

Please refer to the Supplementary file for the justification. Therefore, the cost function is mathematically represented as:

$$\mathbb{C}_{g_i} = C_{g_i} \left\lceil \frac{F_{g_i} S_{g_i}^{max} t_{q,g_i}^{max}}{\alpha S_{g_i}^{t} F_{g_i}^{max} t_{q,g_i}} \right\rceil \tag{5}$$

We map the strategic interactions among the FNs using a game-theoretic approach, where the FNs act as players. These players resolve their price for the services, based on their state. The devices interact among themselves as they have incomplete information about one another. Therefore, we map these interactions among the FNs with an oligopoly based game-theoretic framework.

***Definition* 4.** *We compute the utility function of the i^{th} FN with respect to the utilities of other devices from the difference of the price set by the FN, $\mathbb{P}_{g_i}$ and the cost re-*

quired for performing the allotted task, $\mathbb{C}_{g_i}$. Mathematically, $u(g_i, g_{-i}) = \mathbb{P}_{g_i} - \mathbb{C}_{g_i}$.

Algorithm 1 Price Set by Fog Nodes

INPUTS: $S_{g_i}^{res}$, $F_{g_i}^{norm}$, and $t_{g_i}^{wait}$.
OUTPUT: $\mathbb{P}_{g_i}$: Price for resources of the fog nodes.

```
for i = 1 to M do                ▷ M : Total no. of fog nodes
    Compute cost price, C_{g_i}, according to Equation (5)
    Determine P_{g_i}, and share among other fog nodes
end for
for i = 1 to M do
    Compute utility according to Definition (4)
    Maximize utility by solving Equation (7)
end for
The i^th FN set final price, P_{g_i}.
```

Theorem 1. *The utility function of the i^{th} fog node is concave in nature and represented as $u(g_i, g_{-i}) = \mathbb{P}_{g_i} - \mathbb{C}_{g_i}$.*

Please refer to the Supplementary file for the proof.

Theorem 2. *There exists an equilibrium condition for the utility function of the i^{th} FN, considering the residual storage, computation capability, and price set by the i^{th} FN as constant, such that*

$$U(\mathbb{P}_{g_i}, S_{g_i}^{t}, S_{g_i}^{t-1}, \mathbb{C}, t_{q,g_i}^{*}) \geq U(\mathbb{P}_{g_i}, S_{g_i}^{t}, S_{g_i}^{t-1}, \mathbb{C}, t_{q,g_i}) \quad (6)$$

Proof: In our problem scenario, we consider a co-operative dynamic pricing game to set a price among the FNs for their services. The dynamic price set by these FNs depends on their utility function, which changes with the dynamic nature of their configuration states. Therefore, the optimization function is mathematically represented as,

$$\underset{t_{q,g_i}}{argmax} \quad \left(\mathbb{P}_{g_i} - C_{g_i} \left\lceil \frac{F_{g_i} S_{g_i}^{max} t_{q,g_i}^{max}}{\alpha S_{g_i}^{t} F_{g_i}^{max} t_{q,g_i}} \right\rceil \right) \quad (7)$$

subject to $\mathbb{P}_{g_i} \leq \mathbb{P}_{g_i}^{max}$, $S_{g_i}^{t} \leq S_{g_i}^{max}$, $t_{q,g_i} \leq t_{q,g_i}^{max}$, $F_{g_i} \leq F_{g_i}^{max}$, and $(\mathbb{C}_{g_i}, \mathbb{P}_{g_i}, S_{g_i}, F_{g_i}, \mathbb{P}_{g_i}) \geq 0$.

In order to simplify Equation 7, we apply Lagrangian function given in Equation 8, where $\lambda_1, \lambda_2, \lambda_3$, and λ_4 are the *Lagrangian* multipliers, such that $(\lambda_1, \lambda_2, \lambda_3, \lambda_4) \geq 0$.

$$\mathbb{L}_{\lambda} = \mathbb{P}_{g_i} - C_{g_i} \left\lceil \frac{F_{g_i} S_{g_i}^{max} t_{q,g_i}^{max}}{\alpha S_{g_i}^{t} F_{g_i}^{max} t_{q,g_i}} \right\rceil - \lambda_1(\mathbb{P}_{g_i}^{max} - \mathbb{P}_{g_i}) + \lambda_2(S_{g_i}^{max} - S_{g_i}^{t}) + \lambda_3(t_{q,g_i}^{max} - t_{q,g_i}) - \lambda_4(F_{g_i}^{max} - F_{g_i}) \quad (8)$$

Further, to find an optimal solution to the optimization function, we apply the *Karush-Kuhn-Tucker* (KKT) conditions [20]. The *dual feasibility* and *complementary slackness* conditions are given in Equations 9 and 10, respectively.

$$\nabla_{t_{q,g_i}} \mathbb{L}_{\lambda} = C_{g_i} \left\lceil \frac{F_{g_i} S_{g_i}^{max} t_{q,g_i}^{max}}{\alpha S_{g_i}^{t} F_{g_i}^{max}} \right\rceil - \lambda_3 \quad (9)$$

$$\lambda_i(Y_i) = 0, \text{ and } \lambda_i \geq 0 \quad (10)$$

Further, Y_i represents the constraints of Equation (7), where $i = \{1, 2, 3, 4\}$.

On solving the KKT conditions given in Equations 9 and 10, we obtain the optimal value of t_{q,g_i}, where the utility function attains the maximum value. Therefore, the optimal waiting time is represented as:

$$t_{q,g_i}^{opt} = \sqrt{\left\lceil \frac{F_{g_i} S_{g_i}^{max} t_{q,g_i}^{max}}{\alpha S_{g_i}^{t} F_{g_i}^{max}} \right\rceil \frac{C_{g_i}}{\lambda_3}} \quad (11)$$

Hence, an equilibrium condition exists for an optimal value of waiting time of the FNs. ■

After the price for the resources $\mathbb{P}_{g_i}$ of the FNs are set, the UEs $U = \{u_1, u_2, u_3, ..., u_n\}$ within a cluster initiate their decision making process to offload their respective tasks. Further, any i^{th} FN may serve multiple UEs simultaneously. However, the selection of the UE who offload first to any FN is a crucial issue to resolve. To overcome this problem, we apply the *Single Leader Multi-Follower Stackelberg* game-theoretic approach. The UEs that have common FNs in their vicinity form *clusters* among themselves to share relevant information. The UEs only share messages related to the game. Any other form of critical information or data is not shared among the UEs to protect the security and privacy issues. The UE with the higher configuration acts as the *leader*, and the other UEs act as *followers* within that cluster. The players' strategy is to bid the price for the resources p_{u_i}, according to the quantity of resources required.

Algorithm 2 Decision Making

INPUTS: $\mathbb{P}_{g_i}$ (from Algorithm 1), I_i, and D_i.
OUTPUT: Decision to offload or not and to whom to offload.

```
offload_flag = 0
for i = 1 to N do                ▷ N : Total no. of user entities
    Compute T_i^dev, E_i^dev
    Compute Z_i^dev as in Equation (1)
    for j = 1 to M do
        Compute Z_j^fog as in Equation (2)
        if Z_j^fog ≤ Z_i^dev then
            Assign overheads in H_i[j] = Z_j^fog
            Set offload_flag = 1
        end if
        if Z_j^fog ≤ min_overhead then
            Set H_i[j] = 1 and all others 0
        end if
    end for
end for
if offload_flag = 1 then
    Offload task to fog node with H_i[j] = 1
end if
```

***Definition* 5.** *Desired Resource (R_{u_i}): The desired resource is defined as the amount of resource for which the other UEs present within the FNs' proximity is not depleted. Therefore, we attach a penalty factor with the chunk of resources demanded by that UE. The desired resource of the i^{th} UE is mathematically represented as, $R_{u_i} = R_i\left(1 - \sum_{i=1}^{n} R_i\right)$, where R_i denotes the fraction of resources demanded by u_i.*

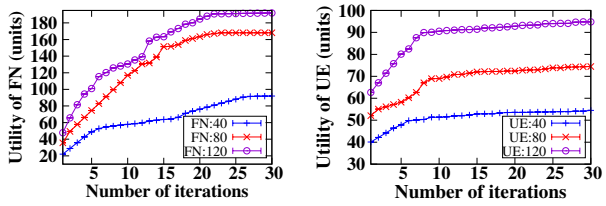
Theorem 3. *The presence of the penalty factor $\left(1 - \sum_{i=1}^{n} R_i\right)$ restricts the i^{th} UE from accessing the entire storage space R_{u_i} available to the nearest FN, g_i.*

Please refer to the Supplementary file for the proof.

***Definition* 6.** *Offered Price (P_{u_i}): The amount offered by the UEs possesses different valuations for the same resource demanded by them. Therefore, we consider another parameter, $\mathfrak{R}$, which is attached to the normalized amount. Mathemati-*

cally, $P_{u_i} = \Re(p_{u_i}/p_{u_i}^{max})$*, where* p_{u_i} *is the amount offered by the* i^{th} *UE, and* $p_{u_i}^{max}$ *represent the maximum amount possessed by that UE.*

Further, we compute the utility function for the UE, u_i, with respect to the other UEs present in the same cluster using Definitions (5) and (6). Therefore, the utility function of the UE is represented as, $\Pi_i(u_i, u_{-i}) = P_{u_i} + R_{u_i}$.



(a) Fog Nodes (b) User Entities

Fig. 4: Utilities for FNs and UEs

Theorem 4. *There exists an equilibrium condition for the utility function of a UE,* u_i*, for a constant amount offered by the UEs. Mathematically,*

$$U(R^*, P) \geq U(R, P) \tag{12}$$

Proof: In our problem scenario, multiple UEs demand for their required resources to the FNs within their proximity. We map this interaction between the UEs and FNs with a co-operative dynamic Stackelberg game. Further, the dynamic bidding of the UEs depend on their utility function, which vary with the fluctuation in demand. Therefore, the optimization function is represented as:

$$\underset{R_i}{argmax} \quad \Re \frac{p_{u_i}}{p_{u_i}^{max}} + R_i\left(1 - \sum_{i=1}^{n} R_i\right) \tag{13}$$

subject to $\sum_{i=1}^{n} R_i < 1$, $p_{u_i} < p_{u_i}^{max}$, and $\Re, p_{u_i}, R_i > 0$. To simplify Equation 13, we apply Lagrangian function as given in Equation 14.

$$\mathbb{L}_\lambda = \Re \frac{p_{u_i}}{p_{u_i}^{max}} + R_i\left(1 - \sum_{i=1}^{n} R_i\right) - \lambda_1\left(1 - \sum_{i=1}^{n} R_i\right) - \lambda_2(p_{u_i}^{max} - p_{u_i}) \tag{14}$$

where λ_1 and λ_2 denote the *Lagrangian* multipliers, such that $(\lambda_1, \lambda_2) \geq 0$. Further, to solve Equation 14, we apply *Karush-Kuhn-Tucker* conditions [20]. The *dual feasibility* and *complementary slackness* conditions are given in Equations 15 and 16. On solving Equations 14, 15 and 16, we obtain the optimal value of resources available.

$$\nabla_{R_i}\mathbb{L}_\lambda = \lambda_1\left(\sum_{j=1, j\neq i}^{n} R_j\right) + 1 - 2R_i - \sum_{j=1, j\neq i}^{n} R_j \tag{15}$$

$$\lambda_i(B_i) = 0, \text{ and } \lambda_i \geq 0 \tag{16}$$

where B_i represents the constraints of Equation (13) and $i = \{1, 2\}$. The optimal solution obtained is mathematically represented as:

$$R_i^{opt} = \frac{1}{2}\left(1 + \lambda_1 \sum_{j=1, j\neq i}^{n} R_j\right) \tag{17}$$

Therefore, we conclude that there exists a Nash equilibrium for the proposed scheme, DEFT. ■

It may be noted that the FNs in an IoV environment serve requests from UEs with varying configurations. Some of the UEs may entirely take up the resources available in an FN. In such a case, the FNs need to deny service requests from the other UEs present in the network. The penalty function in Theorem 3 helps in reducing the possibility of such situations and ensures fairness among the UEs. Intuitively, this fairness may cause a contradiction of interest for the price set by the FNs. However, a Nash equilibrium exists for the proposed system (Theorem 4), which implies that both the UEs and FNs are satisfied with the outcomes. They have no affinity for changing their decisions, which proves the stability of DEFT.

Algorithm 1 provides a comprehensive view of the price set by the FNs. Based on the cost function designed, residual storage, computation capability, and waiting time, the cost price of the FNs is estimated in Step 2. Further, in Steps 5-8, each FNs' utility is computed, and the optimal value of waiting time is derived. Finally, the price is set by the i^{th} FN in Step 9. On the other hand, Algorithm 2 determines whether the UE offloads their task to the FN present within their vicinity or not. Depending upon the overhead incurred for computation done locally and at the fog layer, as in Steps 2 - 14, the offloading decision is taken. Additionally, the hash table is updated, and FN selected for computation offloading is mentioned in Steps 12 and 17. Since Algorithm 1 needs information from all FNs in the network, and calculate the cost and utility for determining the FN's resource price, it has a running time of $\mathcal{O}(\mathcal{M}) + \mathcal{O}(\mathcal{M}) \approx \mathcal{O}(\mathcal{M})$. On the other hand, Algorithm 2 needs $\mathcal{O}(\mathcal{MN})$ time for making offloading decisions. In summary, DEFT asymptotically needs $\mathcal{O}(\mathcal{MN})$ time for making its decisions.

IV. Performance Evaluation

In this section, we evaluate the performance of our proposed scheme. First, we describe our simulation design and then present our observations on running the experiments. We then present how DEFT performs in comparison to the state-of-the-art solutions.

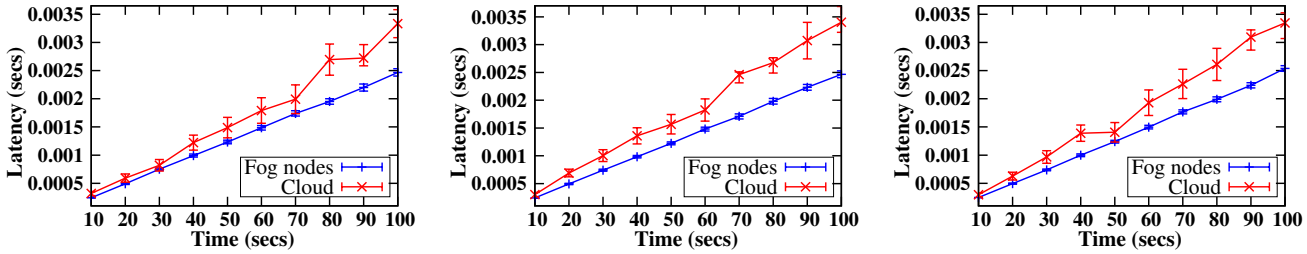
A. Simulation Design

In this section, we evaluate the performance of the proposed scheme DEFT in the presence of 5-120 IoT UEs and 5-80 FNs. We consider an IoV environment, where the UEs are randomly distributed over a geographical region of area $10 \times 10\ km^2$. To compute the speed (S_n) and direction (D_n) of these UEs, we apply *Gauss-Markov* mobility model.

$$S_{n,i} = \alpha S_{(n-1),i} + (1-\alpha)\bar{S}_i + \sqrt{(1-\alpha^2)}S_{r_{(n-1)},i} \tag{18a}$$

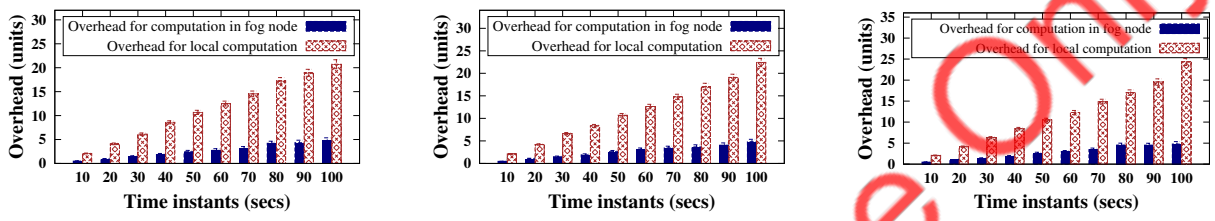
$$D_{n,i} = \alpha D_{(n-1),i} + (1-\alpha)\bar{D}_i + \sqrt{(1-\alpha^2)}D_{r_{(n-1)},i} \tag{18b}$$

where $S_{(n-1),i}$ and $D_{(n-1),i}$ represent the speed and direction of the UEs at the previous time instant $(t-1)$. α represents the tuning parameter, $\bar{S}_i$ and $\bar{D}_i$ denote the mean value of speed and direction of the i^{th} UE. Finally, $S_{r_{(n-1)},i}$ and $D_{r_{(n-1)},i}$ represent the Gaussian random variables of speed and direction. In this work, we set $\alpha = 0.1$ with D_n and S_n in the range of $(0, 100)$ degrees and $(0.8, 1.2)$ Kmph and obtain the trace in Fig. 7(a) and their corresponding velocity in Fig. 7(b). We deploy the FNs randomly around the UEs



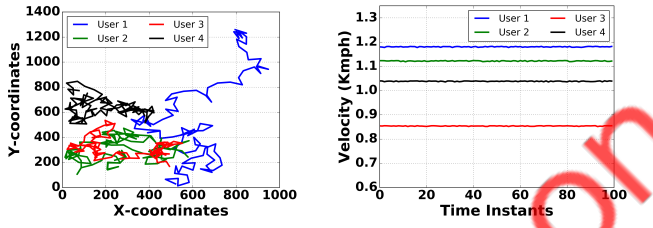
(a) 40 User Entities (b) 80 User Entities (c) 120 User Entities

Fig. 5: Overhead endured when tasks are offloaded to fog nodes and when computed locally



(a) 40 User Entities (b) 80 User Entities (c) 120 User Entities

Fig. 6: Latency in case of fog and cloud computing schemes



(a) Mobility trace (b) Velocity

Fig. 7: Gauss-Markov mobility model

TABLE I: Simulation Parameters

Parameter	Value
FN computation capability	$20-25\ GHz$
FN storage capacity	$1000-2000\ GB$
Shannon's capacity ($r(\Re[t], P_{tx})$)	$30\ KBps$
Drain efficiency (η)	15.7%
Path-loss exponent (α)	2
Constant value (ξ)	0.0005
Power consumed (P_{T_0})	$15.9mW$
Cloud computation capability	$100GHz$

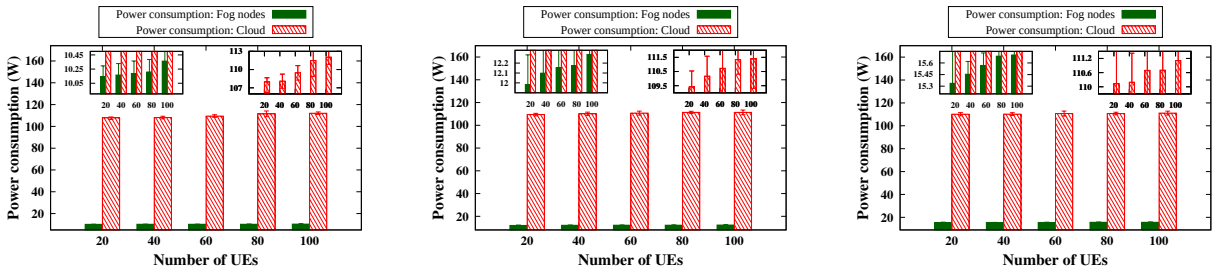
in Fig. 7(a). It may be noted that we illustrate only 4 UEs in Fig. 13 for simplicity. In our simulation, we deploy the FNs and UEs as shown in Table I. We study the overhead experienced by each of the UEs over different time instants. As the rate at which the UEs request for services varies with time, we consider *Poisson distribution* to model their arrival rate at the FNs. We consider the computation capability, storage capacity, and battery capacity of the user entity as 8 — $18GHz$, 8 — $64GB$, and 3 — $5Ah$, respectively. Further, in the simulation, we consider the UEs' task size as 400 — $500KB$. We consider that these tasks execute in 1000 — 2000 cycles. We present the details of the simulation parameters in Table I.

B. Results

In this section, we first describe the parameters latency, power, and throughput, along with a few others, and then analyze the results obtained during the simulation.

Utility: We prove that our proposed game-theoretic solution DEFT attains equilibrium by Theorems 2 and 4. Figs. 4(a) and 4(b) illustrate the variations in the utility of FNs and UEs with the number of iterations. We apply Equations (7) and (13) to compute the utilities achieved by the FNs and UEs. We observe that an increasing trend exists in the utilities of the FNs and UEs, which is due to the maximization of the utility functions. On average, the utilities of FNs converge after 20 iterations, while in the case of UEs, the utilities show convergence after 10 iterations. Such convergence in the utility values implies that the devices do not deviate from their decisions. These utilities account for the device configurations and the tasks from the user entities. With the results in Fig. 4 and Theorems 2 and 4, we conclude that the proposed solution achieves an optimal solution for both the fog nodes and user entities. Additionally, the utility increases as the number of devices (UEs and FNs) increases, which proves that our solution is beneficial for both the consumers and service providers.

Overhead: We describe overhead as the cost (inclusive of device parameters) incurred by the UEs to perform a particular task. We estimate the overhead for computation done locally



(a) 40 Fog Nodes (b) 60 Fog Nodes (c) 80 Fog Nodes

Fig. 8: Power consumption in case of fog and cloud computing schemes

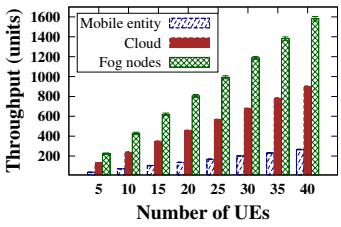


Fig. 9: Throughput in case of Mobile User Entities, Cloud and Fog nodes

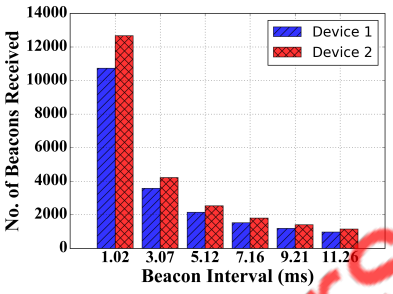


Fig. 10: Beacons received on changing intervals

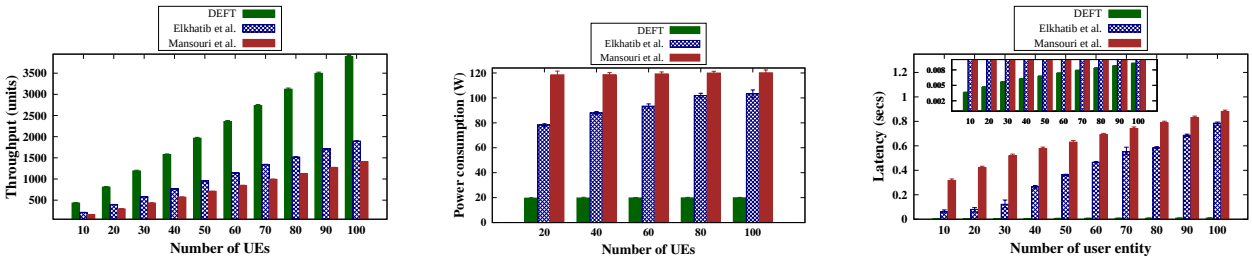
and at the FNs using Equations (1) and (2). In Figs. 5(a)–5(c), we observe that the overhead increases linearly with time. The possible reason behind such a trend is that the FNs are getting busier, and the UEs keep generating tasks that need computation, therefore overhead increases. However, we observe that the overhead for local computation is always higher than the overhead for computation in an FN, which is the main motivation behind our work. Additionally, we observe no significant change in the overhead as the number of UEs increases.

Latency: We compute latency as the time required by the UEs to execute their task. This time is inclusive of the time required to offload the entire task, its execution, and the delivery of results. Figs. (6(a))–(6(c)) show the latency in case of FNs and cloud, in the presence of 40, 80, and 120 UEs. Along the x-axis, we increase the time from 10 up to 100 seconds, with a step of 10. We consider the cloud as a centralized entity and randomly place the FNs. Similar to the trend observed in the case of overhead, the latency also increases with time. We characterize the variations in latency with FNs executing other tasks as new requests arrive from the UEs. However, we notice a few spikes in the latency in the case of the cloud. The probable reason for these variations in latency is the variation in the task size and time required for computation. The latency is reduced by 25.88% on an average at the 100^{th} time instant in case of FNs compared to the cloud.

Power Consumption: Motivated by the works of Roy *et al.* [21], we compute the power consumption, in case of cloud and FNs, as $P_{T_i} = P_{T_0} + \frac{\xi \times d_{ij}^{\alpha}}{\eta} + \frac{I_i \times d_{ij}}{r(\Re[t], P_{tx})}$, where P_{T_i} denotes the power consumed by the i^{th} FN/cloud. P_{T_0} is the power consumed by the transmission circuit, and d_{ij} is the Euclidean distance between the UEs and their nearest FNs/cloud. Figs. 8(a)–8(c) illustrate the variations of power consumption of the UEs using the proposed scheme, DEFT, in the presence of 40, 60 and 80 FNs. The UEs do not possess any choice in offloading to the cloud, as it is considered a centralized entity. However, in the case of FNs, the UEs select the nearest FN and offload their task as we randomly place the FNs close to them. Therefore, it reduces the power consumed significantly. The rate of increase in power consumption with the rise in UEs in case of 40 FNs is about 2.09 %, while in the case of the cloud is 3.67 %, respectively.

Throughput: Throughput is the number of tasks executed locally or remotely per unit time. Fig. 9 shows the variations of throughput in the case of UEs, cloud, and FNs. With the increase in the number of UEs along the x-axis, we observe that an increasing trend exists in the three instances. However, as the FNs spend more time in execution than the transfer of data to and from the UEs, the FNs possess the highest throughput.

Beacons: Fig. 10 depicts the beacons received by two arbitrarily chosen UEs. We vary the beacon intervals to study the behavior of the received packets. We observe in all cases that device 2 receives a larger number of beacons than device 1. We attribute this behavior to the varying distance and motion of the devices/UEs. As we increase the intervals, we observe a decreasing trend in the received beacons due to a decrease in the number of broadcasts. However, with an increase in beacon intervals, the UEs no longer have updated FNs' information. Decisions based on 11.26 ms beacon intervals may result in heavy penalties as the FNs may be in another state. The UEs lose the opportunity to take access to the FNs due to non-updated information. Although the standard beacon interval of 1024 micro-seconds/1.02 ms increases channel and message overhead, it facilitates real-time decisions. We conclude that the lower beacon intervals



(a) Throughput (b) Power Consumption (c) Latency

Fig. 11: Comparison of various parameters with the increase in user entities

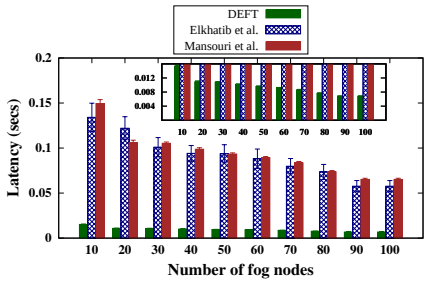
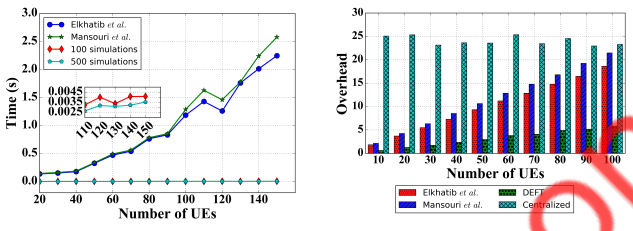


Fig. 12: Comparison of latency with increase in fog nodes



(a) Simulation time (b) Overhead

Fig. 13: Comparison of simulation time and overhead with increasing number of UEs

are beneficial, which results in efficient channel utilization.

C. Benchmark Solution

To evaluate the proposed scheme, DEFT, we perform a comparison with existing benchmark solutions proposed by Mansouri *et al.* [9] and Elkhatib *et al.* [8]. The authors in [9] proposed a resource allocation approach to the IoT users in a hierarchical fog-cloud computing system. They illustrated the delay experienced by each task on processing locally and at the cloud-server, which is equivalent to our proposed scheme's latency. In another work, Elkhatib *et al.* [8] worked on the feasibility of micro-clouds, designed with a collection of Raspberry Pis. We observe the variations in the throughput, power consumption, latency achieved, simulation time, and overhead, with the increase in the number of UEs.

Fig. 11(a) shows that the throughput is increased by 51.06% and 60.57% in our proposed scheme, compared to the existing schemes [8], [9]. We attribute this behavior to the linking of the tasks to the appropriate FNs. DEFT increases the idle FNs utilization and minimizes the burden (impending task queue) from the overloaded ones. Further, in Fig. 11(b), we observe that the power consumed in the case of DEFT is reduced by 75.19% and 79.17% compared to the existing schemes [8], [9]. We calculate the energy consumption using the Equation mentioned in Section IV, which considers the power consumption of the transmission circuit and the Euclidean distance between the UEs and the FNs. Accordingly, we attribute the low power consumption of DEFT to the selection of optimal nearby FNs. We comment that due to the proposed utility function, in addition to the availability of resources, DEFT also considers the selection of the closest favorable FN among the proximal ones. The selection of nearby FNs reduces the energy required for transmission, which reduces the overall energy consumption. We envision the utility of the cloud for storage only. Fig. 11(c) illustrates that DEFT reduces latency significantly. Fig. 12 shows that our proposed scheme, DEFT outperforms existing schemes with an increase in the number of FNs. Interestingly, we observe that with the increase in the number of FNs, the latency follows a decreasing trend. On the other hand, with the rise in the number of UEs, latency follows an increasing trend. We also observe that the rate of increase/decrease in latency, in case of an increase in FNs/UEs, is much lower in amplitude in our case. To examine the real-time processing capability of the proposed scheme, we analyze the computational complexity in terms of the simulation time required to execute the algorithms. Fig. 13(a) demonstrates the variations in the simulation time with the increase in the number of user entities (UEs). We observe that the simulation time of the proposed scheme is reduced by 93% and 92% compared to the existing schemes [8], [9]. Additionally, we observe that the rate of rise in simulation time is significantly low in the case of 500 simulation runs, compared to 100 simulation runs. Fig. 13(b) demonstrates the overhead in our proposed scheme, centralized fog environment, and the existing schemes. We observe that the overhead increases with the increase in the number of UEs; however, the pattern in the increase of overhead differs. In the case of centralized computation offloading, the overhead incurred is always high because it is done without selecting the appropriate FNs. Further, in the existing schemes, the hierarchical fog-cloud [9] and micro-cloud based [8] platform incur higher overhead compared to DEFT. Therefore, we infer from the above discussion that the proposed scheme, DEFT outperforms the other schemes in terms of throughput, power consumption, latency, overhead, and simulation time.

V. Conclusion

In this paper, we proposed a decentralized, two-level game-theoretic solution for computation offloading in an IoV environment. Towards this, we considered the presence of heterogeneous mobile IoV UEs, FNs/RSUs, and the cloud as external platforms for computation offloading. Additionally, we proved the existence of equilibrium for both the UEs and FNs in the game, along with its concavity. Further, we analyzed DEFT with the increase in the number of devices and existing solution approaches, in terms of throughput, power consumption, latency, overhead, and simulation time.

We assumed that the IoV UEs have one task to offload to the FNs/cloud at any particular time instant. In the future, we plan to extend our work by breaking these tasks into smaller subtasks with dependencies by forming directed acyclic task graphs. We also plan to minimize both latency and power consumption by introducing parallelism.

References

[1] R. Kumar, N. K. Chilamkurti, and B. Soh, "A Comparative Study of Different Sensors for Smart Car Park Management," in *Proceedings of The International Conference on Intelligent Pervasive Computing (IPC)*, Oct. 2007, pp. 499–502.

[2] A. Fascista, G. Ciccarese, A. Coluccia, and G. Ricci, "Angle of Arrival-Based Cooperative Positioning for Smart Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 9, pp. 2880–2892, Nov. 2018.

[3] J. Du, L. Zhao, X. Chu, F. R. Yu, J. Feng, and C. I, "Enabling Low-Latency Applications in LTE-A Based Mixed Fog/Cloud Computing Systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1757–1771, Nov. 2019.

[4] H. Cao and J. Cai, "Distributed Multiuser Computation Offloading for Cloudlet-Based Mobile Cloud Computing: A Game-Theoretic Machine Learning Approach," *EEE Trans. Veh. Technol*, vol. 67, no. 1, pp. 752–764, Jan. 2018.

[5] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct 2016.

[6] N. Heuveldop, "Ericsson Mobility Report," *Ericsson, Stockholm, Sweden*, no. EAB-17:005964, Jun 2017.

[7] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast, 2016–2021," *Cisco*, no. C67-738469-00, Jun. 2017.

[8] Y. Elkhatib, B. Porter, H. B. Ribeiro, M. F. Zhani, J. Qadir, and E. Rivière, "On Using Micro-Clouds to Deliver the Fog," *IEEE Internet Computing*, vol. 21, no. 2, pp. 8–15, Mar. 2017.

[9] H. Shah-Mansouri and V. W. Wong, "Hierarchical Fog-Cloud Computing for IoT Systems: A Computation Offloading Game," *IEEE Internet Things J.*, May 2018.

[10] Y. Lin, E. T. . Chu, Y. Lai, and T. Huang, "Time-and-Energy-Aware Computation Offloading in Handheld Devices to Coprocessors and Clouds," *IEEE Syst. J.*, vol. 9, no. 2, pp. 393–405, Jun. 2015.

[11] A. Samanta and Z. Chang, "Adaptive Service Offloading for Revenue Maximization in Mobile Edge Computing With Delay-Constraint," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3864–3872, Apr. 2019.

[12] V. C. Emeakaroha, K. Fatema, L. v. d. Werff, P. Healy, T. Lynn, and J. P. Morrison, "A Trust Label System for Communicating Trust in Cloud Services," *IEEE Trans. Services Comput.*, vol. 10, no. 5, pp. 689–700, Sep. 2017.

[13] E. Meskar, T. D. Todd, D. Zhao, and G. Karakostas, "Energy Aware Offloading for Competing Users on a Shared Communication Channel," *IEEE Trans. Mobile Comput.*, vol. 16, no. 1, pp. 87–96, Jan. 2017.

[14] X. Tao, K. Ota, M. Dong, H. Qi, and K. Li, "Performance Guaranteed Computation Offloading for Mobile-Edge Cloud Computing," *IEEE Wireless Commun. Lett.*, vol. 6, no. 6, pp. 774–777, Dec. 2017.

[15] Z. Zhou, P. Liu, J. Feng, Y. Zhang, S. Mumtaz, and J. Rodriguez, "Computation Resource Allocation and Task Assignment Optimization in Vehicular Fog Computing: A Contract-Matching Approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3113–3125, Apr. 2019.

[16] Z. Zhao, S. Bu, T. Zhao, Z. Yin, M. Peng, Z. Ding, and T. Q. S. Quek, "On the Design of Computation Offloading in Fog Radio Access Networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 7136–7149, Jun 2019.

[17] K. Lei, M. Du, J. Huang, and T. Jin, "Groupchain: Towards a Scalable Public Blockchain in Fog Computing of IoT Services Computing," *IEEE Trans. Services Comput*, vol. 13, no. 2, pp. 252–262, Apr. 2020.

[18] Q. Fan and N. Ansari, "Towards Workload Balancing in Fog Computing Empowered IoT," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 1, pp. 253–262, Mar. 2020.

[19] X. Chen, "Decentralized Computation Offloading Game for Mobile Cloud Computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

[20] M. S. Bazaraa, *Nonlinear Programming: Theory and Algorithms*, 3^{rd} ed. Wiley Publishing, 2013.

[21] A. Roy, P. Kar, S. Misra, and M. Obaidat, "D3: Distributed approach for the detection of dumb nodes in wireless sensor networks," *International Journal of Communication Systems*, 01 2014.

Pallav Kr. Deb is a Institute Research Scholar and pursuing his PhD from the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India. His current research interests include Internet of Things, Cloud and Fog Computing, THz Communications, UAV networks, and Wireless Body Area Networks.

Chandana Roy is a Institute Research Scholar and is pursuing her PhD from the Department of Industrial and Systems Engineering, Indian Institute of Technology Kharagpur, India. Her current research interests include Industrial Internet of Things, Wireless Body Area Networks, and Cloud Computing. She is a student member of the IEEE.

Arijit Roy is a Council of Scientific & Industrial Research Senior Research Fellow (CSIR-SRF) and pursuing his PhD from the Indian Institute of Technology Kharagpur, India. His current research interests include Internet of Things, Cloud Computing, and Wireless Body Area Networks. He is student member of the IEEE. Mr. Roy received several academic awards such as Samsung Innovation Award and IBM Day Award.

Dr. Sudip Misra is a Professor with the Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur. Dr. Misra is the Associate Editor of the IEEE Transactions Mobile Computing and IEEE Systems Journal, IEEE Transactions on Sustainable Computing, IEEE Network, and Editor of the IEEE Transactions on Vehicular Computing. His current research interests include algorithm design for emerging communication networks and Internet of Things.