Edge Selection Non-Cooperative Game in IoT Edge Computing

Kinda Khawam[1,5], Hussein Taleb[2], Hassan Fawaz[3], Samer Lahoud[4], Dominique Quadri[5], and Steven Martin[5]
[1] Universit de Versailles Saint-Quentin-en-Yvelines, 78000 Versailles, France
[2] Ecole Supérieure d'Ingénieurs de Beyrouth, Saint Joseph University of Beirut, Beirut, Lebanon
[3] SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, 91120 Palaiseau, France
[4] Dalhousie University, Halifax, Nova Scotia, B3H 4R2, Canada
[5] ROCS, LISN, Universit Paris Saclay, 91190 Gif-sur-Yvette, France

***Abstract*—Computational offloading is a pivotal solution to several Internet of Things (IoT) issues as it helps subdue the constrained nature of IoT devices. By harnessing the large capacity at the Edge, IoT devices with limited battery and storage can delegate certain tasks, especially those related to Machine Learning. Because of their restricted capacity, such devices can only store a limited amount of data as a training set for their learning, leading to a faulty prediction with high error rate. To tackle that issue, IoT devices can federate the learning process with other devices while the Edge server acts as an aggregator. However, selecting the appropriate Edge is a significant challenge. In fact, although learning collectively can reduce the prediction error, it also brings about a communication cost that depends on the selected Edge. Therefore, in this paper, we propose a Non-Cooperative game where devices autonomously and efficiently select an Edge server in order to reduce both their learning error and communication cost.**

***Index Terms*—Edge Computing, Non-cooperative game theory, Federated learning, Linear regression, IoT.**

I. Introduction

With the dawn of the Internet of Things (IoT), massive data is collected by constrained devices that cannot be put to use to leverage Machine Learning (ML) because of IoT devices' limited resources. Hence, such devices usually transmit the local data they collected to Edge servers to train efficiently inference ML models. However, this inflicts unwarranted computation, storage and communication costs on IoT devices that also suffer from privacy leakage hazards. Federated Learning (FL) [1] tackles these significant challenges by enabling the aggregation of machine learning models, pre-trained on diverse IoT devices, and collectively improving a global model. At the start of each training round, the Edge server distributes the current global model to all participating IoT devices. These devices train their individual local models using their own limited datasets and only transmit model parameter updates to an aggregator at the Edge server upon completing each training round. This iterative process continues until the global model reaches the desired level of accuracy.

However, for FL to give off satisfying performances, it needs to abide by some assumptions of which the two most important ones are the following:

- The first assumption asserts that the data samples detected by various IoT devices represent independent and identically distributed (i.i.d.) random variables. In this work, building on the model in [2], we relax a bit this assumption as it usually does not hold, since the local dataset of a single IoT device may not be representative of the overall population distribution. Such realistic relaxation leads to accounting for the Mean Square Error of the prediction accuracy that no longer equates to zero.
- Second, we need to assume that the size of local datasets generated across federated learners is roughly the same to produce balanced distributions. We keep this assumption as this disparity in dataset sizes is primarily due to the different types of IoT devices and different application scenarios, while we consider in this work a homogeneous scenario with the same type of IoT devices using the same application.

Moreover, merging models via FL diminishes the variance in prediction error by leveraging a broader dataset, yet it elevates the error bias due to the heterogeneity in data. Hence, the learning performances depends on the number of IoT devices that selected the same Edge server. Further, increasing the number of federated learners has another downside as it increases the communication cost among learning devices. Finally, another parameter that needs to be factored in is the channel data rate that directly relates to the selected Edge server. Thus, devices need to select the most suitable Edge server in order to minimize both the learning error and communication cost incurred by federating the learning task. To address those challenges, an Edge server selection problem is tackled as a non-cooperative game by autonomous IoT devices. Accordingly, two types of games are defined depending on the data set characteristics.

The type of federated learning we consider in this work is coined as Edge-enabled in the thorough survey found in [3]. Such a setting avoids resorting to the Cloud for prompt computation and training of the learning model. Federated learning is a perfect fit for edge computing and can leverage both the abundant computation power of edge servers and the data collected on outspread edge devices. Such combination is even more relevant in the presence of constrained IoT devices where the limited computing and storage capability of IoT devices can be a hindrance to the final ML model performance.

In the state of the art, the work in [4] and [5] explored

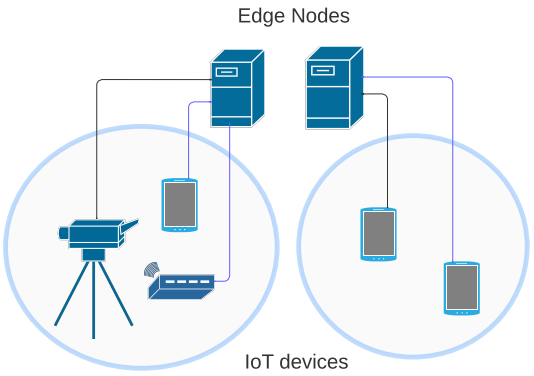

Figure 1: Edge Selection Game

the trade-off between communication cost and the convergence rate of the FL algorithm. Likewise, the work in [6] dealt with resource allocation problem that minimizes the total energy consumption and FL completion time in wireless IoT networks. The work in [7] tackled the client selection for FL in IoT edge computing when the data set are non-i.i.d. as we did in this paper but in the scope of game theory. Many works in the literature have considered FL through the lens of non-cooperative game theory. However, the latter framework is mainly used to incentivize end-users to participate in the Federated Learning process ([8], [9], [10] and [11]). In this work, we address a multi-faceted problem in the context of IoT Edge computing for FL where constrained IoT devices with dissimilar data characteristics need to choose the most suitable Edge server to strike a good balance between efficiency and prompt convergence while taking into account the impact of the wireless channel. We build on the work in [12] but in the latter, we did not account for the data rate impact which is paramount.

The rest of the paper is organized as follows. We provide in section II the system model. In section III, we portray the Edge selection problem as a non-cooperative game and identify two types of games depending on the data set characteristics. For each type of game, we characterize several pure Nash Equilibriums. In section V, we provide practical numerical examples to conduct a detailed analysis, and we subsequently conclude the paper in section VI.

II. The System Model

Federated Learning is particularly suitable when the data is naturally distributed among various IoT devices and collecting that data on a cloud server to reduce the learning error can be costly. Our work goes a step further and tackles the issue of the Edge server selection through which the FL is performed. Accordingly, the devised cost function includes the Mean Square Error (MSE) of the adopted regression learning model explained in II-A and the congestion cost of the Edge server selection detailed in II-B.

A. Error Learning Model

We consider a set $\mathcal{L} = \{1, .., M\}$ of M constrained IoT devices that need to perform ML tasks. Typically, constrained IoT devices sense metrics such as temperature, humidity, etc. and can predict future values for these metrics. Therefore, linear regression is well suited as a ML model and is adopted in this work. Each device j has access to n samples, with θ_j representing the predicted variable, *i.e.*, the coefficient on the true classification function. Following the model outlined in [2], the parameter θ_j for any device j is drawn from a normal distribution with a mean of 0 and variance σ^2, while μ_e denotes the expectation of the error parameter. The training dataset, consisting of n samples, is denoted by X_j, and the predicted variable Y_j is linearly dependent on X_j, expressed as $Y_j = \theta_j \cdot X_j + \eta_j$, where η_j is a random variable with a mean of μ_e. Additionally, the data samples X_j are generated from a normal distribution with a mean of 0 and variance of 1. Utilizing the collected data samples, each device j aims to estimate the mean $\hat{\theta}_j$.

Because of the limited amount of data each device has access to and the limited capacity of the learning devices, this model has relatively high error. Thus, devices can federate the learning process through an Edge server E_k to reduce the prediction error. We consider a set $\mathcal{K} = \{1, .., K\}$ of K Edge servers. As we consider a cellular IoT network, an Edge server is typically the base station. Any IoT device j that selected an Edge server E_k sends to the latter its estimated parameter $\hat{\theta}_j$. Edge server E_k sends back a single model to all devices that selected it as follows:

$$\hat{\theta}_{E_k} = \frac{1}{|\mathcal{E}_k|} \sum_{j \in \mathcal{E}_k} \hat{\theta}_j, \tag{1}$$

where $\mathcal{E}_k$ is the set of devices federating the learning through Edge server E_k and $|\mathcal{E}_k|$ is the cardinal of $\mathcal{E}_k$ such as:

$$|\mathcal{E}_k| = \sum_{j \in \mathcal{L}} \mathbb{1}_{\{device\ j\ selected\ E_k\}}$$

which is the number of devices that selected Edge server E_k.

Further, we define the mean number of devices that selected Edge E_k simply as:

$$\overline{|\mathcal{E}_k|} = \mathbb{E}\Big(\sum_{j \in \mathcal{M}} \mathbb{1}_{\{device\ j\ selected\ E_k\}}\Big) \tag{2}$$

$$= \sum_{j \in \mathcal{M}} \mathbb{E}\big(\mathbb{1}_{\{device\ j\ selected\ E_k\}}\big) \qquad = \sum_{j \in \mathcal{M}} x_j^k \tag{3}$$

where x_j^k is the probability of choosing Edge E_k by device j.

As proven by [2], the MSE recorded for a device federating learning through linear regression with $\overline{|\mathcal{E}_k|} - 1$ other devices, is given by:

$$MSE_k = \frac{\mu_e}{(n-2) \cdot \overline{|\mathcal{E}_k|}} + \sigma^2 \cdot \frac{\overline{|\mathcal{E}_k|} - 1}{\overline{|\mathcal{E}_k|}}, \tag{4}$$

Accordingly, we denote by MSE_k the Mean Square Error recorded by devices learning through the same Edge server E_k.

B. Communication Cost Model

When the learning model is computed through Edge server E_k, each device joining the latter inflicts an additional communication cost among devices in $\mathcal{E}_k$. We adopt a TDMA channel type access like in NB-IoT (Narrow Band IoT) [13] or RedCap [14]. In such a setting, the mean rate of device j when communicating with Edge server E_k is given by:

$$R_{j,k} = \frac{Cap_{j,k}}{\overline{|\mathcal{E}_k|}} \tag{5}$$

where $Cap_{j,k}$ is the mean capacity when device j is communicating alone with E_k and $\overline{|\mathcal{E}_k|}$ is the mean delay endured as devices send in turn (in a given time slot) their estimated parameters to E_k that will operate the aggregated estimation. We consider that one packet is sufficient to send the learned parameter (one metric is learned) and is sent in one time slot. Further, orthogonal channels are re-allocated among Edge servers to cancel out interference. Thus, the communication cost of device j via E_k, denoted by $\delta_{j,k}$, mirrors the delay endured by device j when communicating with the selected Edge server E_k and is hence inversely proportional to the rate obtained:

$$\delta_{j,k} = \frac{\overline{|\mathcal{E}_k|}}{Cap_{j,k}} \tag{6}$$

C. Cost Function

Accordingly, the cost of a device j through Edge E_k is as follows (for $x_{j,k} = 1$):

$$C_{j,k} = MSE_k + \alpha\delta_{j,k}, \tag{7}$$

where α is a normalizing factor. We deduce the mean cost of device j:

$$\overline{C_j} = \sum_{k\in\mathcal{K}} x_j^k \cdot \Big(\frac{\frac{\mu_e}{(n-2)} - \sigma^2}{1 + \sum_{i\in\mathcal{M}} x_i^k} + \frac{\alpha}{Cap_{j,k}}(1 + \sum_{i\in\mathcal{M}} x_i^k)\Big), \tag{8}$$

III. Non-Cooperative Edge Selection Game

We define the continuous game $G^c = \left\langle \mathcal{M}, S^c, \overline{C} \right\rangle$:

- The strategy of device $j \in \mathcal{M}$, denoted by $x_j = \{x_j^k\}, \forall k \in \mathcal{K}$, where x_j^k is the probability that device j chooses Edge server E_k.
- The space of strategies formed by the Cartesian product of each set of strategies $S^c = S_1^c \times ... \times S_M^c$. Therefore, $S_j^c = [0,1], \forall j \in \mathcal{M}$.
- A set of cost functions $\overline{C} = \{\overline{C_1}, ..., \overline{C}_M\}$ given by (8).

In particular, we will consider a more realistic setting where we restrict the choice of any device to its two closest Edges (or Base Stations). Hence, only devices that select the same couple of Edges, deemed $S_l = \{E_l, E_l'\}$, engage in the same non-cooperative game $G_l = \left\langle \mathcal{M}_l, S^l, \overline{C}^l \right\rangle$, where $\mathcal{M}_l$ is the set of IoT devices that compete for the same Edges E_l and E_l'. Hence, the strategy of device $j \in \mathcal{M}_l$, coined x_j^l, is the probability of choosing Edge E_l doted with a data rate of Cap_j (hence, choosing E_l' with probability $1 - x_j$ through capacity Cap_j'). The strategy space of device j is $S_j^l = [0,1], \forall j \in \mathcal{M}_l$.

Finally, the cost function of device j simplifies from (8) to:

$$\overline{C}_j^l(x_j^l, x_{-j}^l) = x_j^l \cdot (F_j(x_{-j}^l) - H_j(x_{-j}^l)) + H_j(x_{-j}^l) \tag{9}$$

where

$$F_j(x_{-j}^l) = \frac{\frac{\mu_e}{(n-2)} - \sigma^2}{1 + \sum_{\substack{i\in\mathcal{M}_l \\ i\neq j}} x_i^l} + \frac{\alpha}{Cap_j} \cdot \Big(\sum_{\substack{i\in\mathcal{M}_l \\ i\neq j}} x_i^l + 1\Big) \tag{10}$$

and

$$H_j(x_{-j}^l) = \frac{\frac{\mu_e}{(n-2)} - \sigma^2}{M_l - \sum_{\substack{i\in\mathcal{M}_l \\ i\neq j}} x_i} + \frac{\alpha}{Cap_j'} \cdot (M_l - \sum_{\substack{i\in\mathcal{M}_l \\ i\neq j}} x_i) \tag{11}$$

with $M_l = |\mathcal{M}_l|$.

Proposition 1 *The $G_l = \left\langle \mathcal{M}_l, S^l, \overline{C}^l \right\rangle$ has a pure NE.*

Proof: For every $j \in \mathcal{M}_l$, the set S_j^l is compact, and $\overline{C}_j^l$ is strictly convex w.r.t. x_j^l (as it is linear in x_j^l) and continuous w.r.t. $x_i^l, i \neq j$. Hence, a Nash equilibrium exists according to [15]. Further, best response dynamics permit attaining the NEs.

A. Computing Mixed Nash Equilibirums

We will compute in this section the various Nash Equilibriums.

Proposition 2 *The Nash equilibrium is either the solution of the following system of equations:*

$$\frac{\beta}{Y_j + 1} + \frac{\alpha}{Cap_j} \cdot (Y_j + 1) = \frac{\beta}{M_l - Y_j} + \frac{\alpha}{Cap_j'} \cdot (M_l - Y_j), \forall j \in \mathcal{M}_l \tag{12}$$

where $Y_j = \sum_{\substack{i\in\mathcal{M}_l \\ i\neq j}} x_i^l$ and $\beta = \frac{\mu_e}{(n-2)} - \sigma^2$.
Or at the boundaries of the strategy space.

Proof: Since the cost functions are convex, at the Nash equilibrium, the $x_j^l, \forall j \in \mathcal{M}_l$ are obtained by computing the partial derivative of the cost function in (9) of each player in respect to its action x_j^l and by equating the result to zero.

$$\frac{\partial \overline{C}_j^l}{\partial x_j^l} = F_j(x_{-j}^l) - H_j(x_{-j}^l) = 0, \forall j \in \mathcal{M}_l$$

When (12) gives a real solution (in fact (12) is a polynomial of degree 3 in Y_j with one guaranteed real solution) such as $Y_j = K_j, \forall j \in \mathcal{M}_l$ we can deduce the embedded x_j^l given by:

$$x_j^l = \frac{\sum_{i=1}^{M_l} K_i}{M_l - 1} - K_j \tag{13}$$

However, the solution in (13) is not always feasible (not necessarily between 0 and 1). Therefore, we will study in the next section the existence of Pure Nash Equilibriums (PNE).

B. Computing Pure Nash Equilibirums

In this section, x_j^l are binary variables that equal one if device j selects Edge server E_l and 0 if Edge server E_l' is selected.

The cost function of device j can be reformulated as:

$$C_j^l = \begin{cases} \frac{\beta}{|E_l|} - \sigma^2 + \frac{|E_l|}{Cap_j} & \text{if} \quad x_j^l = 1 \\ \frac{\beta}{|E_l'|} - \sigma^2 + \frac{|E_l'|}{Cap_j'} & \text{if} \quad x_j^l = 0 \end{cases} \tag{14}$$

where $\beta = \frac{\mu_e}{(n-2)} - \sigma^2$, $|E_l|$ (resp. $|E_l'|$) is the number of devices that chose Edge server E_l (resp. E_l'). Recall that $M_l = |E_l| + |E_l'|$.

Moreover, we need to distinguish two cases: case I where $\beta = \left(\frac{\mu_e}{n-2} - \sigma^2\right) \leq 0$ and case II where $\beta = \left(\frac{\mu_e}{n-2} - \sigma^2\right) > 0$. In each case, we will compute the Pure Nash Equilibriums.

1) PNE for Case I: We compute the discrete derivative of the cost function in (14) relative to the number of devices that selected the same Edge server:

$$dC_j^l = \begin{cases} -\frac{\beta}{|E_l|(|E_l|-1)} + \frac{\alpha}{Cap_j} & \text{if} \quad x_j^l = 1 \\ -\frac{\beta}{|E_l'|(|E_l'|-1)} + \frac{\alpha}{Cap_j'} & \text{if} \quad x_j^l = 0 \end{cases} \tag{15}$$

It is straightforward to verify that the cost function is increasing in the number of devices that chose the same Edge server as $\beta \leq 0$. Hence, we are in presence of the so-called unweighted crowding game. In such games, the cost function is player specific (as it depends on the mean capacity of the IoT device) and it is non-decreasing in the number of players that selected the same strategy. According to [16], unweighted crowding games with only two strategies (Edge servers E_l and E_l') have the Finite Improvement Property (FIP). Interestingly, PNE of games with the FIP can be attained with fully distributed Replicator dynamics [17].

Let *mixed strategy* $q_j = (q_{j,1}, q_{j,2}, \ldots, q_{j,K})$ be a probability distribution over pure strategies. In other words, pure strategy $a_j(t) = s$ is chosen with probability $q_{j,s} \in [0,1]$, with $\sum_{s=1}^{K} q_{j,s} = 1$. Let Q_j be the simplex of mixed strategies for IoT device j. Let $Q = \prod_{j=1}^{M} Q_j$ be the space of all mixed strategies. A *strategy profile* $Q = (q_1, ..., q_M) \in Q$ specifies the (mixed or pure) strategies of all players. Following classical convention, we write $Q = (q_j, Q_{-j})$, where Q_{-j} denotes the vector of strategies played by all other devices.

Definition 1 The game mechanics work as follows: at $t = 0$, we begin with $q(0) = (q_1(0), ..., q_M(0))$ any random vector of probabilities. At each iteration $t > 0$:

1) Each device j chooses an action $a_j(t)$ (aka an Edge node) according to probability distribution $q_j(t)$.
2) Each device j learns the cost $C_j(t)$ resulting from its choice $a_j(t)$ and the set of all actions of other players.
3) Each device j updates the probability vectors $q_j(t+1)$ in the following way:

$$q_{j,s}(t+1) = \begin{cases} q_{j,s}(t) + b(1 - \frac{C_j(t)}{C_{max}})(1 - q_{j,s}(t)) \\ \text{if} \quad s = a_j(t), \\ q_{i,s}(t) - b(1 - \frac{C_j(t)}{C_{max}})q_{i,s}(t) \\ \text{otherwise,} \end{cases} \tag{16}$$

where $0 < b < 1$ is a parameter and C_{max} is the maximal cost, obtained when all devices select the same Edge.

Theorem 1 *Consider G as an instance of the game $\mathcal{G}$. Let Q^* denote a set of mixed profiles where at most one player employs a pure strategy. Irrespective of the initial condition within $Q - Q^*$, the learning algorithm invariably weakly converges to a Nash Equilibrium.*

The convergence proof for Theorem 1 can be found in [17].

These types of algorithms are fully distributed, as decisions made by devices are entirely decentralized: at any iteration t, device j only needs to know the cost $C_j(t)$ of selected Edge node, which in turn depends on the number of devices that selected that same Edge node and the the corresponding data rate. One message sent by the Edge node to its active devices and communicating the number of those devices will suffice as a signaling message. That signaling message can be sent as a unicast message by the Edge node to any device that selected it anew.

2) PNE for Case II: For Case II, we resort to an efficient heuristic to pinpoint PNE. We know that at Nash Equilibrium, all devices are in mutual best response, denoted by $z_j^l, \forall j \in \mathcal{M}_l$. Hence, according to (9), the optimal response of device j satisfies what follows:

$$z_j^l(t+1) = \begin{cases} 0, \text{ if } F_j(z_{-j}^l(t)) > H_j(z_{-j}^l(t)), \\ 1, \text{ if } F_j(z_{-j}^l(t)) < H_j(z_{-j}^l(t)), \\ \text{Select randomly } \{0,1\}, \text{ if } F(z_{-j}^l(t)) = H(z_{-j}^l(t)) \end{cases} \tag{17}$$

Thus, we can build on (17) to conceive an iterative search algorithm that is easy to implement. With a starting point $x_j(0)$ for $j \in \mathcal{M}_l$, the algorithm iterates until the strategies in the previous iteration ($x_j^l(t-1)$) and the current iteration ($x_j^l(t)$) remains the same for all devices: Extensive simulations carried out show the convergence of the proposed Algorithm 1 to Pure Nash Equilibriums.

Algorithm 1 Iterative Best Response Dynamics to reach PNE

Initialize Let $x_j(0)$ can be any vector of probabilities $\forall j \in \mathcal{M}_l$.
repeat
  For $j = 1, ..., M'$
    1) Compute $F(x_{-j}(t-1)) - H(x_{-j}(t))$
    2) Set $x_j(t)$ according to (17)
until *All devices have the same strategy as in the previous round*;
Output $z_j^l = x_j^l(t)$ for all $j \in \mathcal{M}_l$.

IV. Price of anarchy

To evaluate the efficiency of the non-cooperative game approach, we compare it against the optimal solution where

we seek to minimize the total cost of IoT devices. The corresponding optimizing problem ($\mathcal{P}$) can be written as follows:

$$\underset{\boldsymbol{a}}{\text{minimize}} \quad C_{tot}(\boldsymbol{a}) = \sum_{j=1}^{M} C_j(a_j, a_{-j}) \tag{18}$$

$$\text{subject to} \quad a_j^k = \{0, 1\},\ \forall j \in \mathcal{L},\ \forall k \in \mathcal{K} \tag{19}$$

$$\sum_{k=1}^{K} a_j^k = 1,\ \forall j \in \mathcal{L} \text{ and } \sum_{j=1}^{M} a_j^k > 1,\ \forall k \in \mathcal{K} \tag{20}$$

where $\boldsymbol{a} = \{a_j, j = 1, .., M\}$.

The problem ($\mathcal{P}$) represents a binary non-linear optimization problem. While such problems can theoretically be solved using exhaustive search algorithms, the computational complexity is in $O(M^K)$. This renders exhaustive search computationally demanding and intractable even for small-sized networks. Consequently, we turn to Simulated Annealing heuristics (SA) [18] as an alternative approach.

The SA heuristic has an acceptance probability that prevents the algorithm from terminating at a local minima. Moreover, the SA algorithm is quite effective in comparison with the exhaustive search.

Algorithm 2 SA heuristic

Initialize Let $\boldsymbol{a}(t = 0)$ such as devices are spread on Edge nodes.

repeat

For $j = 1, ..., M$

Select randomly an edge node and compute $C_{tot}(t)$

If $C_{tot}(t) < C_{tot}(t-1)$, update $\boldsymbol{a}(t)$

Otherwise, update $\boldsymbol{a}(t)$ with probability $e^{\frac{C_{tot}(t)-C_{tot}(t-1)}{T}}$.

++t;

until $t < N$;

Our heuristic algorithm begins with an initial feasible solution, distributing all devices evenly across Edge nodes while ensuring each Edge node serves more than one device. Subsequently, in each iteration, a device is randomly selected to shift from its Edge server to the next one. The resulting configuration is considered a candidate solution, and the corresponding total network cost is calculated. If the candidate solution reduces the total cost compared to the previous iteration, it is accepted. Otherwise, acceptance is determined by a given probability. The algorithm continues iterating until reaching a maximum number of iterations. The SA temperature, denoted by T, is maintained throughout the algorithm.

V. Numerical Simulations

In this section, we evaluate the performance of the two different Edge server selection games, $\beta \leq 0$ and $\beta \geq 0$, and compare the cost realized at Pure Nash Equilibriums (PNE) and mixed Nash Equilibriums. We compare these results to the optimal solution using an exhaustive search for a small network and using the Simulated Annealing meta-heuristics.

For simulation, we use MATLAB where we consider 500 simulation snapshots. Each is repeated until convergence is achieved. All performance metrics are averaged and reported with 95% confidence intervals. For both cases, $\beta \leq 0$ and $\beta \geq 0$ respectively, we considered the following parameters:

- Case I: $\mu_e = 30$, $n = 12$, and $\sigma = 3$.
- Case II: $\mu_e = 30$, $n = 12$, and $\sigma = 1$.

A. Case I: $\beta \leq 0$

In Fig. 2, we illustrate the average network cost at Pure NE and the Mixed NE. At PNE, a lower network average cost is achieved compared to the Mixed NE regardless of the number of IoT devices. However, the Replicator Dynamics, coined by RL for Replicator Learning, used to obtain PNE necessitates many iterations to reach convergence as displayed in Fig. 3, whereas the Mixed NE is obtained directly through calculus.

However, the number of iterations required to reach convergence by the RL solution is much higher than that needed by the BR solution, which converges very rapidly within a maximum of 10 iterations for 120 devices. In fact, when RL is applied, the device learns by taking actions and interacting with the environment through feedback in the form of rewards or penalties. This can be done in a distributed fashion by balancing between exploration and exploitation. Thus, the RL requires more iterations than the BR that is typically implemented in a semi-distributed fashion.

In Fig. 4, we evaluate the total network cost obtained by the optimal solution solved via exhaustive search, the SA meta-heuristic, the cost at PNE and Mixed NE. Due to the high computational complexity of the optimal solution, we limited the number of users to 5 and 10. Here again, the mixed NE renders to highest cost. Moreover, as the number of devices increases, the total cost achieves by the SA heuristic and at PNE become very close to that obtained by the optimal scheme.

By increasing the number of devices in the network, we resort only to SA since the exhaustive search is computationally hard. In Fig. 5, we compared the total cost via the SA heuristics and the total cost at PNE and Mixed NE. We note the small discrepancies between results obtained for the sub-optimal SA and at PNE regardless of the number of IoT devices in the network, while performances at Mixed NE constantly lag behind.

B. Case II: $\beta \geq 0$

The same analysis is conducted for the second case. In Fig. 6, we illustrate the average network cost at Pure NE and Mixed NE. At PNE, a lower network average cost is achieved compared to the Mixed NE regardless of the number of IoT devices. Furthermore, the Iterative Best Response Dynamics used to obtain PNE necessitates a few iterations to reach convergence as displayed in Fig. 7, in comparison with the Replicator Dynamics of the first case. In fact, when Replicator Dynamics is applied, the device learns by taking actions and interacting with the environment through feedback in the form

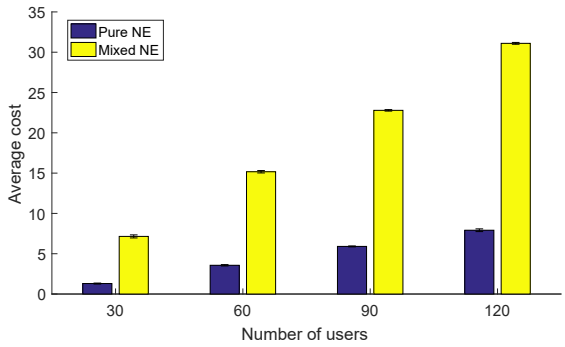

Figure 2: Average network cost vs the number of devices

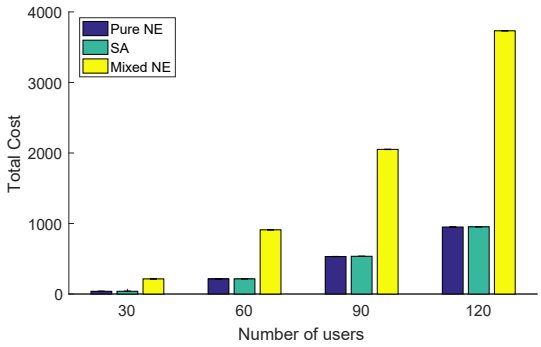

Figure 5: Total network cost vs the number of devices

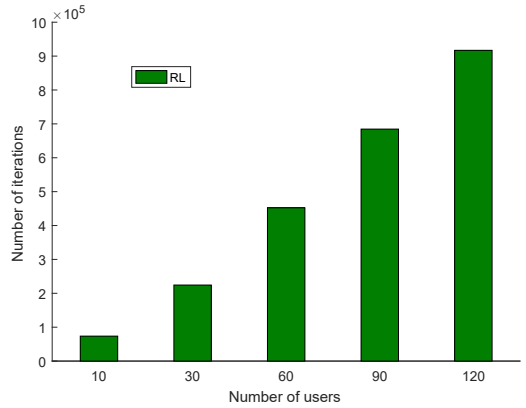

Figure 3: Number of iterations vs the number of devices

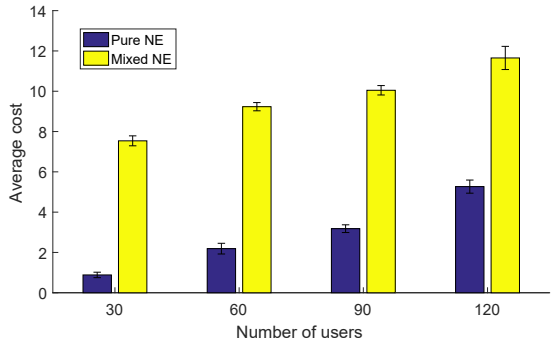

Figure 6: Average network cost vs the number of devices

of rewards or penalties. This is done in a completely distributed fashion by balancing between exploration and exploitation. Thus, the Replicator Dynamics (RL) requires much more iterations than the iterative Best Response Dynamics that is typically implemented in a semi-distributed fashion.

In Fig. 8, we evaluate the total network cost obtained by the optimal solution solved via exhaustive search, the SA meta-heuristic, at PNE and Mixed NE. Due to the high computational complexity of the optimal solution, we limited the number of devices to 5 and 10. Here again, the mixed NE renders the highest cost. The optimal solution gives as expected the lowest total cost, followed closely by the SA heuristics. The distributed approach at NE gives off slightly worse results.

By increasing the number of devices in the network, we resort only to the SA algoritm since the exhaustive search is computationally hard. In Fig. 9, we compared the total cost via the SA heuristics and the total cost at PNE et Mixed NE. Contrary to the first case, we note a wider discrepancy between results obtained for the sub-optimal SA and at PNE. Performances at Mixed NE are still worse off.

We conclude that for both cases, mixed NE can be precluded. Moreover, Pure NE are preferred to mixed ones as they lead to a

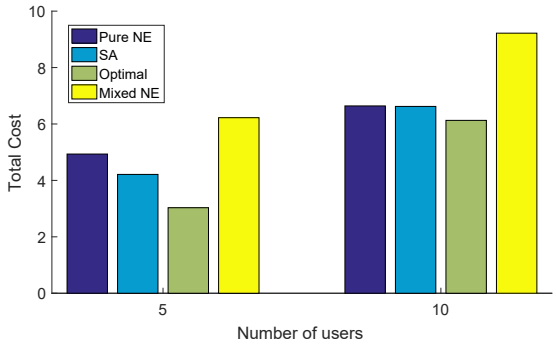

Figure 4: Total network cost vs the number of devices

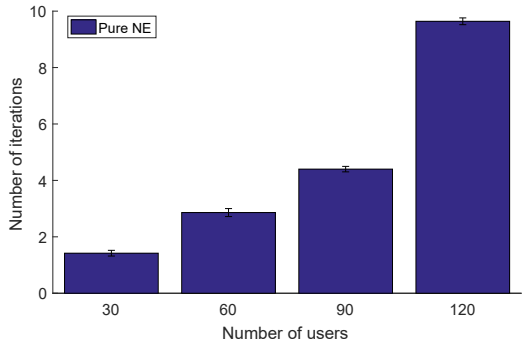

Figure 7: Number of iterations vs the number of devices

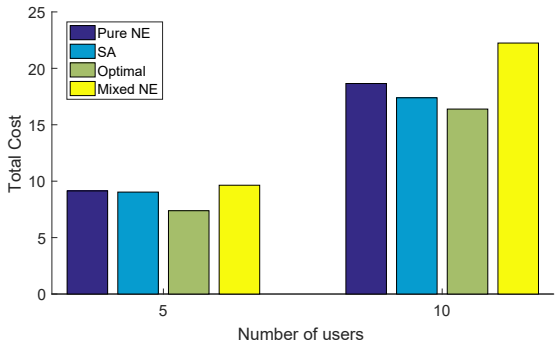

Figure 8: Total network cost vs the number of devices

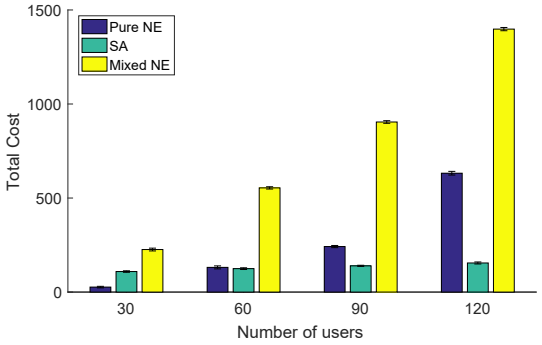

Figure 9: Total network cost vs the number of devices

clear cut Edge selection instead of probabilistically choosing an anchor Edge. Moreover, results at PNE for Case I are very close to the optimal performances at the cost of extended convergence times. As for Case II, convergence to PNE is relatively pretty swift at the cost of sub-optimal performances.

VI. Conclusion

Federated learning offers access to extensive data, a crucial advantage for constrained IoT devices with limited memory. Learning from a larger dataset diminishes variance in the model, thereby decreasing errors. However, federating the learning process involves communication costs among devices, a factor to consider. In this paper, devices engage in a non-cooperative game by selecting an Edge server for federated learning. Autonomous device selections aim to minimize both learning error and communication costs. Conclusions are consolidated through intensive numerical simulations showing that efficient Edge selection is reached at Pure Nash Equilibrium through a fully distributed Replicator dynamics or semi-distributed Best Response dynamics depending mainly on the data set characteristics.

References

[1] Y. Zhan, J. Zhang, Z. Hong, L. Wu, P. Li, and S. Guo, "A survey of incentive mechanism design for federated learning," *IEEE Transactions on Eeing Topics in Computing*, 2021.

[2] K. Donahue and J. M. Kleinberg, "Model-sharing games: Analyzing federated learning under voluntary participation," *CoRR*, vol. abs/2010.00753, 2020.

[3] H. G. Abreha, M. Hayajneh, and M. A. Serhani, "Federated learning in edge computing: A systematic survey," *Sensors*, vol. 22, 2022.

[4] J. Konen, H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.

[5] B. Luo, X. Li, S. Wang, J. Huang, and L. Tassiulas, "Cost-effective federated learning design," 2021.

[6] T. X. V. S. C. JV.-D. Nguyen, S. K. Sharma and B. Ottersten, "Efficient federated learning algorithm for resource allocation in wireless iot networks," in *IEEE Internet of Things Journal*, 2021.

[7] W. Zhang, X. Wang, P. Zhou, W. Wu, and X. Zhang, "Client selection for federated learning with non-iid data in mobile edge computing," *IEEE Access*, vol. 9, 2021.

[8] K. L. U., P. S. Raj, T. N. H., W. Saad, Z. Han, N. M. N.H., and H. C. Seon, *EEE Communications Magazine*, vol. 58, no. 10, 2020.

[9] R. Gupta and J. Gupta, "Federated learning using game strategies: State-of-the-art and future trends," *Computer Networks*, vol. 225, 2023.

[10] J. Weng, J. Weng, H. Huang, C. Cai, and C. Wang, "Fedserving: A federated prediction serving framework based on incentive mechanism." IEEE INFOCOM, 2021.

[11] E. Tahanian, M. Amouei, H. Fateh, and M. Rezvani, "A game-theoretic approach for robust federated learning," *International Journal of Engineering*, vol. 34, no. 4, 2021.

[12] H. F. S. L. D. Q. K. Khawam, H. Taleb and S. Martin, "Non-cooperative edge server selection game for federated learning in iot," *IEEE NOMS*, 2024.

[13] E. M. Migabo, K. D. Djouani, and A. M. Kurien, "The narrowband internet of things (nb-iot) resources management performance state of art, challenges, and opportunities," *IEEE Access*, vol. 8, 2020.

[14] S. Moloudi, M. Mozaffari, S. N. K. Veedu, K. Kittichokechai, Y.-P. E. Wang, J. Bergman, and A. Hglund, "Coverage evaluation for 5g reduced capability new radio (nr-redcap)," *IEEE Access*, vol. 9, 2021.

[15] J. B. Rosen, "Existence and uniqueness of equilibrium points for concave n-person games," *Econometrica*, vol. 33, 1965.

[16] I. Milchtaich, "Congestion Games with Player-Specific Payoff Functions," *Games and Economic Behavior*, 1996.

[17] P. Coucheney, C. Touati, and B. Gaujal, "Fair and efficient user-network association algorithm for multi-technology wireless networks," *In Proc. of the 28th conference on Computer Communications miniconference (INFOCOM)*, 2009.

[18] D. T. Pham and D. Karaboga, "Intelligent optimisation techniques: Genetic algorithms, tabu search, simulated annealing and neural networks," *Springer-Verlag New York, Inc., USA, 1st edition*, 1998.