SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V12I8P117

Volume 12 Issue 8, 188-198, August 2025
© 2025 Seventh Sense Research Group®

Original Article

Mobility -Aware Resource Allocation in Edge Networks for Smart Neighborhoods

Sravya Pallantla[1], D. Haritha[2], Shanti Chilukuri[3]

[1]Department of Computer Science and Engineering, JNTUK, Andhra Pradesh, India.
[2]Department CSE, University College of Engineering, JNTUK, Andhra Pradesh, India.
[3]Department of Computer Science and Engineering, School of Technology, GITAM (Deemed to be University), Andhra Pradesh, India.

[1]Corresponding Author : sravyapallantla@gmail.com

Received: 09 June 2025 Revised: 10 July 2025 Accepted: 11 August 2025 Published: 30 August 2025

Abstract *- Wireless Sensor Networks (WSNs) comprise sensor and actuator nodes that either generate or consume data. These end devices typically have limited memory, energy, and computational capacity, making them ill-suited for intensive processing tasks. Edge computing addresses this limitation by offloading data collection, processing, and forwarding to edge nodes with greater resources. This architecture aligns well with WSNs, but efficient allocation of edge resources remains a significant challenge, particularly under node mobility and fluctuating traffic. The resource allocation problem at the edge is NP-hard, and while static solutions exist, they often fail under dynamic network conditions. Memory management for packet queues is especially critical. The scheduling and dropping policies at edge nodes directly influence Quality of Service (QoS). Weighted Fair Queuing (WFQ), a popular scheduling method, assigns different weights to traffic classes, affecting uplink bandwidth distribution. However, bursty data, varying packet sizes, and node mobility complicate fair and efficient weight assignment. This study proposes a federated learning-based framework for dynamic queue and bandwidth allocation in mobile WSNs. The model adapts to real-time changes in traffic and topology while reducing communication overhead. Simulation outcomes demonstrate improved resource utilization and network performance, validating the effectiveness of the proposed approach in dynamic WSN environments.*

Keywords *- Wireless Sensor Networks (WSN), Resource Allocation, Mobility, Federated Learning.*

1. Introduction

Smart neighborhoods are neighborhoods that have data gathering and/or automation mechanisms for better and sustainable cities, leading to improved quality of living. Wireless Sensor Networks (WSNs) are the key enabling technology for such neighborhoods. These networks are formed by sensor nodes deployed in the neighborhood areas to collect and transfer information, and actuators that can be used for automation. Since the sensors and actuators (the *end nodes*) are characterized by limited energy, bandwidth, and processing power, there is a need to control the usage of the available resources to sustain good performance and longevity of the network.

The data generation patterns in such networks may be periodic with low or high periodicity or event-driven, depending on the application requirements. In Mobile Wireless Sensor Networks (MWSNs, typical in smart neighborhood applications), where either the sensor nodes or base stations are mobile, resource allocation becomes significantly more complex due to frequent changes in network topology. Due to these uncertainties, it is not possible to model such systems accurately. Owing to better computational power and data availability, Machine Learning (ML) has proven to be an attractive approach to study the input-output relations in problems that cannot be modeled well.

While several ML strategies have been applied in the context of resource allocation for wireless and/or mobile networks, two main questions plague the practicability of such approaches:

1. Where is the data for training the ML model gathered?
2. Where is the ML model trained?

Gathering statistically relevant data requires considerable memory. Further, training ML models requires computational power, both of which may be scarce in a WMSN. One way of addressing this resource scarcity is to have edge nodes in the network. Edge nodes transmit data from/to a set of designated end nodes and the core network. Compared to the end nodes, edge nodes typically have better computing power and memory resources. End nodes consume less energy by communicating with the edge nodes instead of the core network itself. In addition, the edge nodes may aggregate the data gathered, reducing the traffic in the core network. They may also act as packet filtering firewalls, ensuring network security.

However, the inclusion of edge nodes in the network introduces the problem of resource allocation at the edge nodes. Especially, efficient utilization of memory at the edge nodes is crucial for reliable and timely transmission of data, as buffer overflows at the edge nodes may result in the loss of critical data. While there are several queuing models that help plan the queue capacity and aid in traffic shaping, they rely on the fact that the packet arrival rates and patterns are known a priori, which may not be the case with WMSNs.

In recent times, Machine Learning has been used for various resource allocation problems at the edge nodes [1]. Edge nodes, while receiving and transmitting the data from the end nodes, gather information about the data arrival patterns and train an ML agent that allocates resources to maximize some network performance parameter.

Performance parameters that are typically of interest and need to be optimized are the overall network efficiency (bandwidth utilization) or the Fairness of allocation in terms of the goodput of each flow of data. Once the ML model is trained, it can be used to infer optimal allocations for time-varying data arrival patterns, which may be due to bursty data generation or the Mobility of the end nodes.

For arriving at an ML model that can recommend optimal resource allocation schemes for unseen network scenarios, a lot of independent and identically distributed (iid) data is required. Edge nodes that are in the vicinity of a set of end nodes may only see certain data patterns and may not be able to arrive at good, generalizable models.

One approach to solve the above problem is to transmit all the data gathered by the edge nodes to a central server, where the ML model is trained. However, this is bandwidth-intensive and may compromise the security of the data [2].

Federated Learning (FL) is an ML paradigm that helps create generalized models even when non-IID data is gathered at distributed locations in the system, while reducing the data transmissions required for creating such models and preserving data security [3]. In this paper, we present a resource allocation strategy for edge nodes in WMSNs with time-variant traffic patterns in Smart neighborhoods, to optimize the Efficiency of allocation or the Fairness of the allocation for different network flows. The strategy is based on Weighted Fair Queueing (WFQ), with the weights for each network flow being determined by an ML model trained using Federated Learning. Our contributions are:

- Design of a WFQ-based algorithm that leverages FL for optimal (in terms of Efficiency or Fairness) resource allocation at the WMSN edge.
- Evaluation of the proposed algorithm (using simulation with well-researched, real-world data generation patterns in smart neighborhood applications) for a wide range of parameters such as different network sizes, mobility patterns and queue lengths.

To the best of our knowledge, the problem of optimal resource allocation at the edge nodes in WMSNs has not been studied with real-world data generation patterns of Smart neighborhoods.

2. Background

2.1. Mobility Models for End Devices

The mobility pattern of the end nodes greatly influences the traffic pattern at the edge nodes and hence plays a major role in optimal resource allocation. The most popular models for node motion are the Random Walk 2D, Random Direction 2D, Random Waypoint, and Gauss-Markov models.

For example, Random Walk and Random Direction are applied to create random motion of nodes, while Gauss-Markov imitates realistic moving patterns of correlated mobile nodes.

The data received at an edge node at any instant depends on the *network scenario*, which comprises the network topology (the links between the end and edge nodes and the quality of such links) and the amount of data generated by each end node at that instant.

2.2. Weighted Fair Queuing

In this section, we very briefly explain how the WFQ discipline works. Though this is an elementary discussion, we include it here for quick reference.

Queue management at the edge nodes comprises two aspects – the scheduling policy and the allocation (including the drop) policy.

While the scheduling policy determines which outgoing packet is sent out to the core network from the packet queue, the allocation policy determines how the incoming packets are stored in the packet queue, following a particular drop policy when the allocated space for a flow is exhausted.

The WFQ scheduling algorithm is extensively used in traffic management, where bandwidth is fairly rationed for different data flows. Two essential components (shown in Figure 1) are necessary for implementing the WFQ discipline [4].

- The WFQ classifier that stores the packets from each flow in the appropriate queue and
- The WFQ scheduler serves the packets according to the minimum virtual finish time.

Schedule Packets from Queues Based on Weight

The weighted queuing approach discussed in this paper aims to ensure consistent throughput for end devices, regardless of the fluctuations in their data rates or the quality of the links connecting them to the edge node [5].

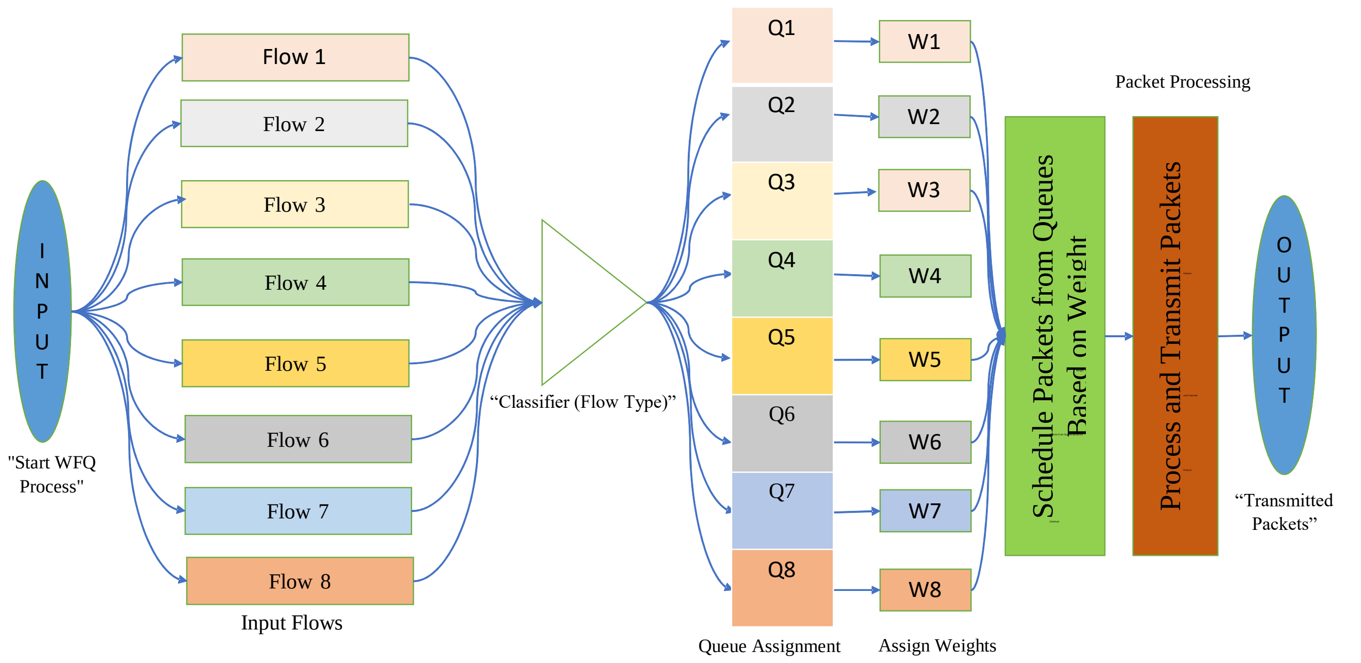


Fig. 1 Weighted fair queue

The WFQ classifier divides traffic flows into two or more classes and assigns a percentage of the available bandwidth to each. The incoming traffic is stored in separate queues, based on its class. A non-negative weight w_i^* is assigned to the i^{th} Queue. As shown in Equation (1), the amount of bandwidth assigned to a queue at any given time is calculated by normalizing the weights corresponding to nonempty queues. w_i^* [6].

$$w_i^* = \frac{w_i}{\sum w_j^*} \tag{1}$$

The value of w_i^*, which varies from 0 to 1, denotes the fraction of the overall bandwidth assigned to queue i. In every t seconds, every nonempty queue sends $\mathcal{B} * t * w_i^*$ Bits on a link of bandwidth $\mathcal{B}bps$.

The WFQ scheduler assigns a start time and an end time to every packet. The first and last bits of the packet are served by the scheduler at these virtual times, respectively. Assuming that the k^{th} packet of flow i, denoted by P_i^k arrives at the time A_i^k. Let its start and finish times be denoted by S_i^k and F_i^k. Equations (2) and (3) define S_i^k and F_i^k [4].

$$S_i^k = max(F_i^{k-1}, A_i^k) \tag{2}$$

$$F_i^k = S_i^k + \frac{\mathcal{L}_i^k}{w_i} \tag{3}$$

Where $F_i^0 = 0$, $\mathcal{L}_i^k$ is the length of the packet P_i^k, and w_i^* Is the weight of flow i? The virtual finish times of the packets determine the order in which they are transmitted, with the scheduler choosing a packet with the shortest virtual finish time to be scheduled next.

A major issue with traffic flows where the traffic pattern is dynamic is the assignment of weights to the different flows. While static weight assignment is alright when dealing with flows that have constant traffic generation patterns, it does not work for end devices that generate data intermittently. Due to this, there has been a lot of research on dynamic weight assignment in WFQ to achieve a given application QoS parameter, such as delay. However, most past work assumes a *proportional relative* QoS model [7]. As per this, each queue is assigned a weight, and the ratios of the QoS parameters achieved for a set of flows are equal to the ratios of their weights. In general, such proposals have an observation period during which the traffic pattern is observed to decide the weight and are generally applicable for flows delivered by fixed, wired networks, such as those for audio/video streams. Smart neighborhoods, on the other hand, may have some mobile devices with intermittent traffic generation patterns and wireless connections. Such networks may take a long time to be observed, and edge nodes in smart neighborhoods may not have enough memory to store traffic and observe a pattern. Our work differs from such work in two major aspects:

- One of our goals is efficient resource (queue memory) utilization, which is different from achieving proportionally relative QoS.
- We use a pre-trained model based on a common set of devices with wireless links and mobility patterns in smart neighborhoods. This reduces the memory and processing requirements at the edge nodes.

2.3. Performance Metrics for Edge Resource Allocation

Resource allocation may have different goals. In this paper, we focus on two different goals: the Fairness of the buffer allocation and the overall Efficiency of allocation. Efficiency in the context of resource allocation refers to how well the available resources (e.g., bandwidth, computational power, or memory) are utilized to maximize system performance. On the other hand, Fairness is concerned with ensuring that all devices receive a proportional share of the available resources, which is especially important in scenarios where devices have varying capabilities or access to resources [8, 9]. At an edge node, Fairness and overall Efficiency may be contradictory goals, and hence, we focus on each of these at a time.

Fairness ensures that resources are distributed equitably among users, tasks, or nodes, especially in a multi-user or multi-node system. In edge computing, particularly with FL, Fairness ensures that no single node or user monopolizes the resources, leading to starvation or degradation of service for others. The fairness F(X) of an allocation is determined by Equation (4). Here, H and L represent the upper and lower limits of QoE, respectively, while σ denotes the standard deviation of the QoE values.

Considering channel allocation among M individuals (different flows), let X = $(x_1,\ x_2 \ldots x_m)$ be the allocation vector, where x_i is the $i^t h$ flow allocation percentage. To quantify Fairness, we draw upon the fairness formula as Equation (4), which is presented in reference [10]. Let σ_x Represent the standard deviation of throughput values arising from allocation X. The upper bound H and lower bound L correspond to the maximum and minimum throughput values attained within allocation X.

$$F(X) = 1 - \frac{2\sigma_x}{H-L} \tag{4}$$

Equation (5) determines the Efficiency, which is the ratio between the input data rate. I_i and the throughput T_i of all flows that arise from the given allocation X as below:

$$E(X) = \frac{1}{m}\sum_{i=1}^{m} \frac{T_i}{I_i} * 100 \tag{5}$$

2.4. Federated Learning

Due to the advances in data storage and computing technology in the past two or three decades, machine learning is being increasingly used for obtaining meaningful inferences from data. While supervised machine learning, where a dataset that includes both inputs and corresponding outputs is used for training models, is a popular choice for data-driven inferences, the main challenge in such approaches is the availability of statistically significant data. Typically, in environments such as smart neighborhoods that use the edge AI model, edge nodes (nodes at the core network edge) gather the data and train ML models for smart network management. However, each edge node sees only the data from the end devices in its network and learning from only this data may not result in inferences that can deal with new network scenarios. To overcome this, in a traditional centralized machine learning approach, the data from *different* edge nodes is sent to a central server, where the training takes place. The resultant model based on the diverse data sample sets is then broadcast to all edge nodes. While this results in a much more general model that can be used for meaningful inferences in new network scenarios, it involves high communication overhead. Such overhead is not suitable for low-power devices and for wireless networks that are characteristically bandwidth(and energy)-poor [11]. One approach that cuts down the cost of data communication for training is federated learning. The idea behind FL is to allow participants to train local models without sharing the local data with a central server [12]. The FL server collects locally trained models from different (edge) nodes and aggregates them to produce a global model. This approach is well-suited particularly for resource-constrained networks, as communicating the updates of the model instead of the whole dataset is considerably cheaper in terms of communication.

```
Algorithm 1: (FL Model N)

Inputs: Local dataset for each node: $D_i$, Number of epochs: E,
Set of computing nodes: N
Output: Federated Averaged Model: $M_{avg}$
On Each Computing Node i:
Input: Current global model $M_{avg}$
Split the local dataset $D_i$ into mini-batches of size $B_i$, each
mini-batch $b \in B_i$
For each mini-batch $b \in B_i$ do
    Train the model using mini-batch data.
    Update the local model. $M_{avg}^{b+1}$ from $M_{avg}^{b}$ using the
mini-batch data
End For
```

```
Send the updated local model $M_{avg}^{i}$ to the Federated Learning (FL) server
On the FL Server:
Initialize the global model $M_{avg}^{0}$
For each epoch j=1 to E do
    For each node k∈N do
        Call Local_Training (k, $M_{k}^{j}$) to obtain the updated local model $M_{k}^{j+1}$
    End For
    Compute the Federated Averaged Model $M_{avg}^{j}$ By averaging the models from all nodes:
        $M_{avg}^{j} \leftarrow \frac{1}{N}\sum_{k=1}^{N} M_{k}^{j+1}$
End For
```

Algorithm 1 shows the FL process. There are two types of participants in the federated learning model - N computing nodes, each denoted by N_i, $i \in [1, N]$and the FL server. In the beginning, the FL server initiates the model M_{avg}^{0}, the learning rate of hyperparameters, and shares this model with all the computing nodes. Each node gathers a local dataset D_i . That is split up into several batches of a size. B_i. For every batch b, the training algorithm will run locally to get the resulting model. M_{avg}^{b+1}. After all the batches of a local set D_i are trained, the resultant model M_{avg}^{i} Will be sent to the FL server by the computing node. The FL server creates an aggregated global model from the models received from computing nodes. N_i, $i \in [1, N]$. However, aggregation of local models just once is often not a good enough solution. The FL server repeats this procedure for some epochs to get the optimal model. M_{avg}^{j} $j \in [1, E]$.

3. Literature Survey

The domain of Wireless Sensor Networks (WSNs) and resource allocation has seen various approaches to improve energy efficiency and network lifetime [13]. Zaman et al. introduced an energy-aware protocol to extend WSNs' lifetime, while Heinzelman et al. [14] proposed the LEACH protocol, which uses hierarchical clustering and cluster head rotation to enhance network longevity. However, traditional methods often fail to address adaptive network topologies and mobility issues effectively.

In the context of Mobility in WSNs, Temene et al. highlighted the unique challenges posed by mobile sensor nodes, such as frequent topological changes and communication overhead [15]. Yan et al. developed energy-efficient protocols that consider node mobility, but centralized algorithms in these solutions often lead to scalability challenges in large-scale mobile environments due to high overhead costs [16]. Machine Learning (ML) has also been explored for resource optimization in WSNs. Researchers [17] have utilized reinforcement learning to manage node operations and reduce energy consumption, as well as machine learning techniques to improve routing efficiency [18]. While ML-based approaches have proven effective in static WSNs, their application in mobile WSNs remains underexplored. Challenges such as the absence of a central database and the need for near-real-time management complicate ML's implementation in mobile environments.

Federated Learning (FL) has emerged as a promising approach for distributed networks. Beltrán et al. [19] proposed FL for decentralized environments where centralized data collection is not feasible, and Li et al. demonstrated its effectiveness in reducing latency and enhancing system efficiency in edge computing [20]. FL offers significant benefits for WSNs, such as decentralized decision-making, energy savings, and data protection. However, its application in mobile WSNs is still limited, with few studies addressing dynamic node mobility and other characteristics of such networks.

In resource allocation for mobile WSNs, FL has shown potential. Studies like those by Yu et al. [6] Yang et al. highlighted FL's utility in optimizing cache usage, managing mobile edge computing resources, and enhancing energy efficiency while minimizing delay in vehicular networks [21]. Despite these advancements, the integration of mobility patterns into FL-based models for resource allocation remains scarce, with most research focusing on static or semi-dynamic networks.

Several challenges and gaps persist in current research. Centralized and distributed resource allocation approaches often face scalability and communication issues in mobile environments. Most ML methods require centralized data, which is increasingly impractical due to privacy concerns. Although FL offers a promising solution, its application in mobile WSNs and the use of mobility patterns for dynamic resource allocation are still underexplored. This paper aims to address these gaps by proposing a novel FL technique for optimal resource allocation in mobile WSNs. The approach leverages node mobility to dynamically allocate resources such as bandwidth and queue, while improving overall performance metrics like energy efficiency and throughput. Through simulation, the proposed method is demonstrated to outperform existing approaches in terms of convergence rate, Fairness, and adaptability to varying network conditions.

4. Resource Allocation for Edge Nodes using FL

For a given set of mobile devices and wireless channel properties (MAC protocol, noise, propagation model, etc.), the network graph varies with the mobility pattern of the devices and the area in which the devices move (the grid). The throughput at the next downstream node (from the edge node) also depends on the link capacity between it and the edge node. Let the total uplink bandwidth available at the edge node be L, with F data flows. The sum of the bandwidth allocated to each data flow A_i [1 to N] must be less than or equal to the total available uplink bandwidth, L. Here A_i It is computed from Equation (3) based on the WFQ used at the edge node.

- Grid Size (G): The spatial dimension of the simulated area.
- Mobility Speed (M): The speed of node mobility in meters per second (m/s).
- Link Capacity (L): The total transmission capacity in Kbps of the link between the edge nodes and the next downstream node.

Figure 2 shows the learning in the edge network. The edge nodes participate in the federated learning to first construct a training model based on the collected (local) data for T seconds, to predict the weights of bandwidth allocation for a specific flow, leading to optimal Efficiency or Fairness.

The FL server constructs a global model by collecting the training models from all such edge nodes. The global model takes into account the data in a network with a specific number and type of devices and parameters (G, M, L) to make dynamic inferences of the Efficiency or Fairness (based on the objective) for each bandwidth allocation case. Using Federated Learning takes into account the different traffic patterns at each edge node to arrive at a more general model that can make better allocation recommendations.

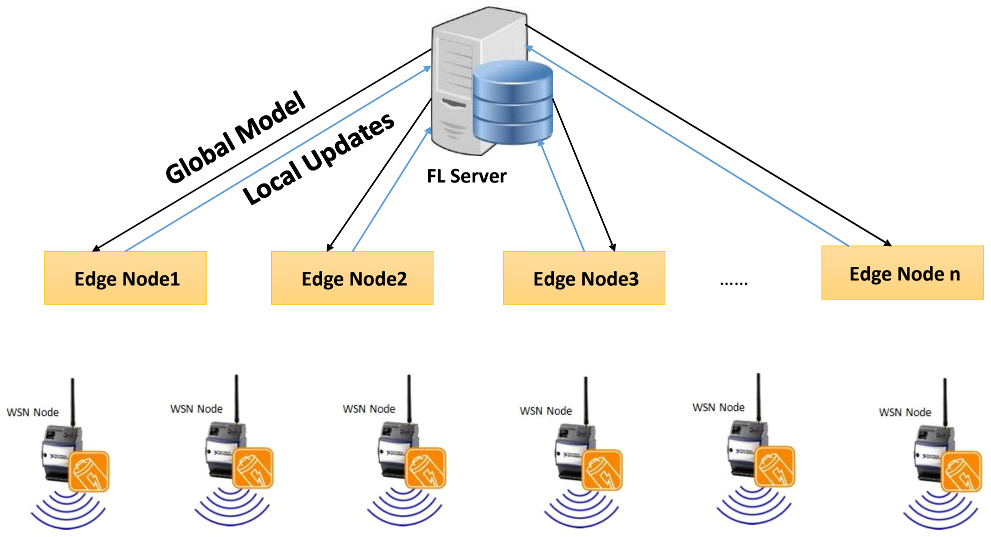


Fig. 2 Federated learning model

```
Algorithm 2: FL_allocation

Inputs: Maximum number of nodes, maxnodes
Outputs: Efficiency ($E_i$) and Fairness ($F_i$) for each configuration
Initialize: nodes ← 1
While nodes ≤ maxnodes do
        Set N←nodes
        Call Algorithm 1 with input N
        Obtain the global model $M_{avg}$ from Algorithm 1
        Apply $M_{avg}$ on inputs (G, M, L)
        Calculate Efficiency $E_i$
        Compute Fairness $F_i$
        Increment nodes ← nodes + 1
end While
```

The Algorithm 2 iteratively evaluates different node configurations for federated learning by varying the number of participating nodes. For each configuration, it computes the resulting global model's Efficiency and Fairness to assess performance scalability.

During the network operations phase, each edge node observes the network parameters and uses the model to predict the Efficiency or Fairness that would result from each allocation case. It then chooses that allocation that maximizes the objective (Efficiency or Fairness).

The proposed solution aims to optimize resource allocation and performance in Smart neighborhood environments by using Federated Learning (FL) with a focus on Fairness and Efficiency.

5. Results and Discussion

Since the FL model requires training data, we used realistic network simulations using ns-3, with varying mobility models and network parameters. The ns-3 simulator

[22] also validates the proposed resource allocation strategy [23].

Table 1. Wireless network parameters

Parameter Name	Value
Frequency Band	802.11 (5 GHz)
Modulation Scheme	16-QAM
Propagation Characteristics	delay: Constant Speed Propagation Delay Model loss: Log Distance Propagation Loss Model
Transmit Power	16 dBm
Antenna Specifications	• Type: Omnidirectional • Gain: 2 dBi
Simulation Software	NS-3
Mobility Model	• Random Walk 2D • Random Direction 2D • Random Waypoint • Gauss-Markov

For simulation, we considered an edge network with each edge node serving up to eight end nodes (some of which are mobile). The edge nodes are connected to end nodes via IEEE 802.11 links. Table 1 shows the parameters used in the simulation. The edge nodes send data to an upstream server via a dedicated, wired link with a data rate L of 1,3 or 5 Kbps. Each end device generates data traffic as shown in Table 2. The data coming from different end devices is classified into different classes based on the type of end device. The data from each end device is relayed to the server by the nearest edge node. In the edge node, the WFQ queuing technique is used to serve the packets based on the weights allocated for each flow. Each end device has D_i its own queue of Q_i Bits at the edge node and the edge node apply the WFQ principle to forward data on the fixed downstream link.

During the data-gathering phase, the deployment area of the network (specified by the grid size G) was varied from 150m to 300m in a step size of 50m. The mobility speed M of each end device was simulated to be from 2m/s to 5m/s in steps of 1m/s, with each of the five mobility models. Each "network scenario" consisted of a fixed grid size, mobility speed, model and set of end devices. For such a scenario, the edge node varied the weight (W_i) of each flow from 0.1 to 0.2 such that the sum of all flow weights is 1, and observed the resulting Efficiency or Fairness. The tuple $(W_1, \dots, W_n, Q_1, \dots Q_n, G, M, L)$, contains the network scenario parameters and the test bandwidth allocation for a given network scenario. This tuple forms the input to the FL agent, and the resulting Efficiency (or Fairness) observed during data gathering is the output. This input and output were recorded every 1000 simulation seconds and added to the training dataset.

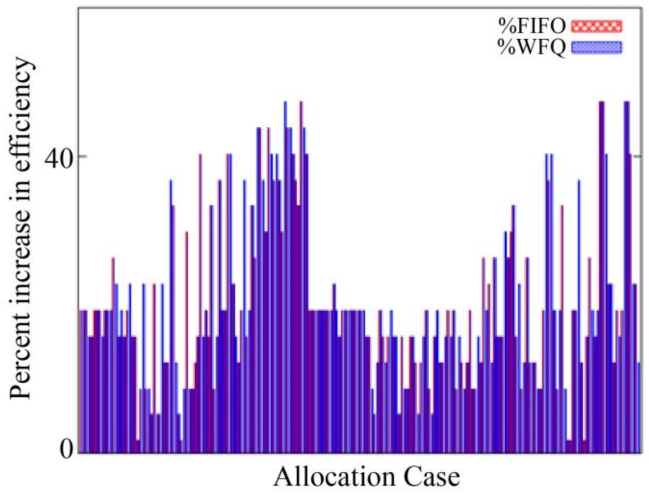
We consider two baseline schemes - one with a simple WFQ with equal weights for each of the flows (0.125 for each of the eight flows) and the scheme described in [3]. The percentage increase in Efficiency (or Fairness) that is predicted by the proposed scheme for each allocation case over each of these schemes is plotted for various network scenarios and mobility models. In each case, the edge node then chooses the allocation case that gives the best benefit in terms of the goal (Efficiency or Fairness) for optimal goal achievement.

Table 2. End devices traffic parameters

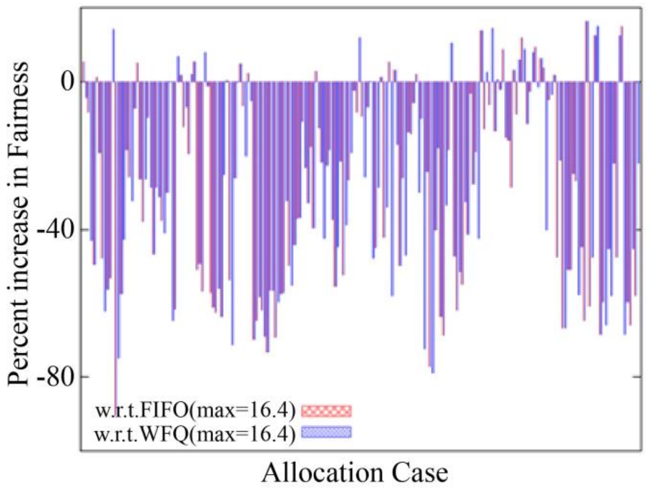
Type of Device	Average Sleep Time (s)	Mean Data Rate (bps)	Packet Length (bits)	Mobility
Smoke Sensor	4	462	234	Yes
Motion Sensor	4	11388	234	Yes
Amazon Echo Hub	4	462	144	No
Smart Phone	4	2461	327	Yes
Smart Plug	241	462	144	No
Smart Bulb	4	462	94	No
Smart Camera	4	2461	234	Yes
Wireless Speaker	4	462	144	Yes

Once the training dataset is collected, the nodes run a federated learning algorithm using the Adam optimizer [24]. The ReLU activation function and mean squared error loss are used for training. We considered 19 features; the neural network consists of a 19 x 64 input layer, 64 x 128, 128 x 256, 256 x 128, 128 x 64 hidden layers, and a 64 x 1 output layer. Early stopping with a patience of 25 epochs was used to prevent overfitting. The average loss was calculated by varying the number of computing nodes from 1 to 3.

Figures 3 visually demonstrate the Efficiency and Fairness across all possible allocation vectors (a total of 206 vectors for 8 flows and quantum values ranging from 0.1 to 0.2, such that the sum of allocated bandwidths is 1.0), corresponding to a specific state denoted as S = (G, M, L). This state is characterized by the grid size, mobility speed, and link capacity of each edge node. The grid size is 200, the speed is 2 m/s, and the link capacity is 5 kbps. All mobile devices move according to the Random Walk 2D mobility model.



(a)

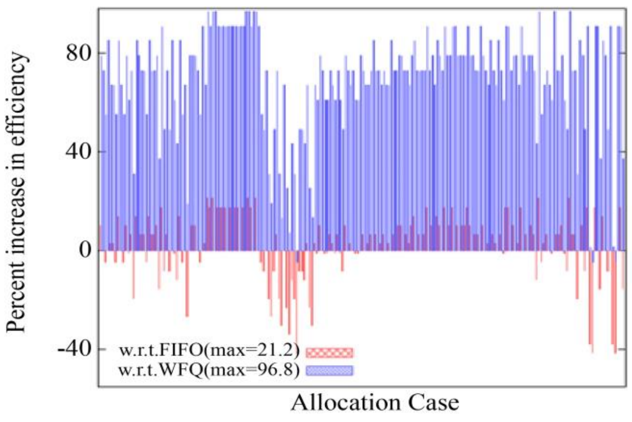


(b)

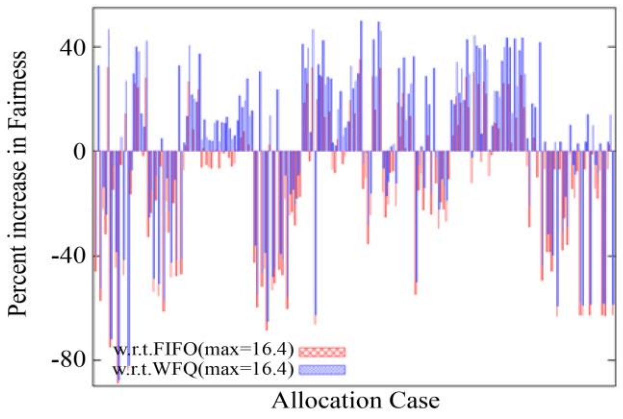
Fig. 3 (a)Comparative analysis of Efficiency for S_1=(200,2,5) for Random walk (2D) Mobility, and (b)Comparative analysis of Fairness for S_1=(200,2,5) for Random walk (2D) mobility.

As shown in Table 3, the maximum Efficiency occurs when the allocation is (0.1, 0.15, 0.1, 0.15, 0.1, 0.1, 0.2, 0.1) for these devices and the mobility model. Table 4 shows that the maximum Fairness occurs when the allocation is (0.1, 0.1, 0.1, 0.2, 0.2, 0.1, 0.1, 0.1), and up to 16% more Fairness can be achieved with the proposed scheme.

Figure 4 shows the Efficiency and Fairness for the 206 different allocation vectors. The grid size considered is 200, speed 2 m/s and link capacity 5 kbps, and the mobile devices move as per the Random direction 2D mobility model. Table 3 shows that the maximum Efficiency occurs when the allocation is (0.1, 0.15, 0.1, 0.1, 0.15, 0.2, 0.1, 0.1) for this network scenario. Up to 96.8% more Efficiency (compared to the simple WFQ scheme) can be achieved with the proposed scheme. The maximum Fairness occurs when the allocation is (0.1, 0.15, 0.2, 0.1, 0.1, 0.15, 0.1, 0.1).



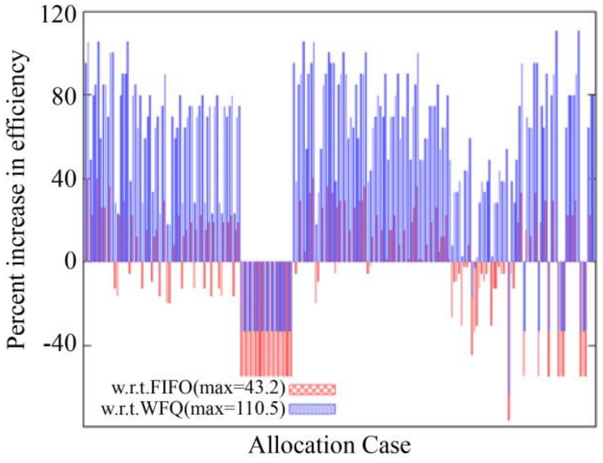
(a)



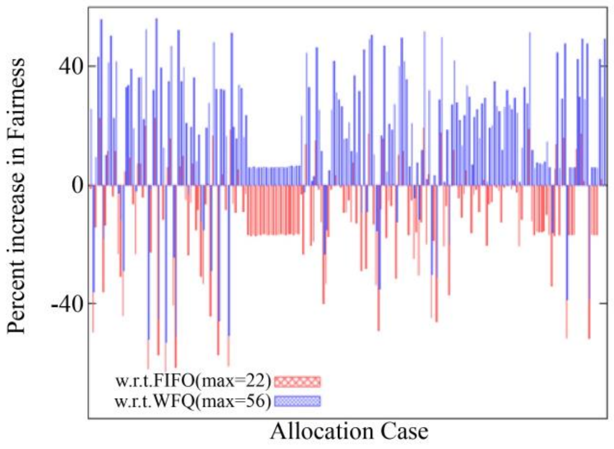
(b)

Fig. 4 (a)Comparative analysis of Efficiency for S_1=(200,2,5) with Random Direction (2D) Mobility, and (b)Comparative analysis of Fairness for S_1=(200,2,5) with Random Direction (2D) Mobility.

Figure 5 illustrates the efficiency and fairness graph for the Gauss-Markov mobility model with the 206 allocation cases. The results highlight how mobility patterns influence Efficiency, requiring careful parameter tuning for optimal performance. The optimal Efficiency, Fairness and the allocation cases leading to these are given in Tables 3 and 4.



(a)



(b)

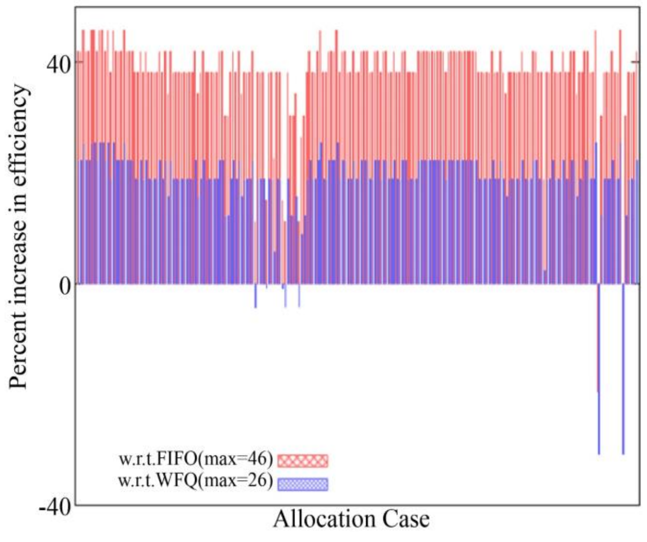
Fig. 5 (a)Comparative analysis of efficiency for S_1=(200,2,5) with Gauss-Markov mobility, and (b) Comparative analysis of fairness for S_1=(200,2,5) with Gauss-Markov mobility.

Table 3. Maximum values of efficiency for different mobility models

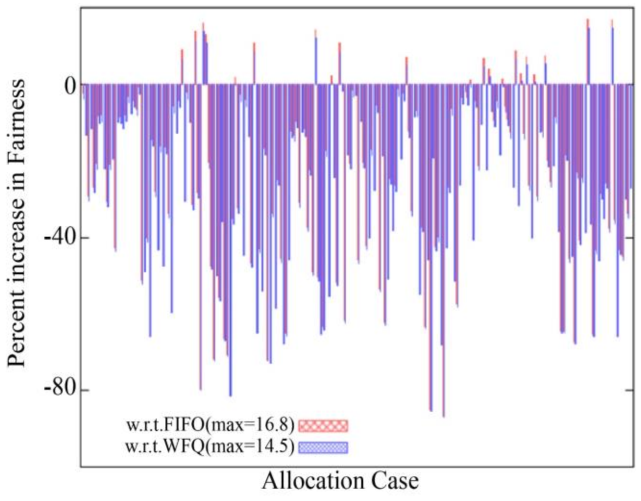
Mobility Model	**Max. Efficiency**	**Allocation leading to Max. Efficiency**	**Simple WFQ**	**FIFO**
Random Walk 2D	28%	(0.1, 0.15, 0.1, 0.15, 0.1, 0.1, 0.2, 0.1)	18%	18%
Random Direction 2D	33%	(0.1, 0.15, 0.1, 0.1, 0.15, 0.2, 0.1, 0.1)	16.9%	27.2%
Gauss-Markov	41%	(0.1, 0.1, 0.1, 0.1, 0.2, 0.2, 0.1, 0.1)	19.5%	28.6%
Random Waypoint	38%	(0.1, 0.1, 0.1, 0.1, 0.1, 0.15, 0.15, 0.2)	30.2%	28.6%

Table 4. Maximum values of fairness for different mobility models

Mobility Model	**Max. Efficiency**	**Allocation leading to Max. Fairness**	**Simple WFQ**	**FIFO**
Random Walk 2D	0.5	(0.1, 0.1, 0.1, 0.2, 0.2, 0.1, 0.1, 0.1)	0.42	0.42
Random Direction 2D	0.5	(0.1, 0.15, 0.2, 0.1, 0.1, 0.15, 0.1, 0.1)	0.33	0.37
Gauss-Markov	0.5	(0.1, 0.1, 0.15, 0.1, 0.15, 0.1, 0.2, 0.1)	0.32	0.408
Random Waypoint	0.49	(0.1, 0.1, 0.1, 0.2, 0.1, 0.1, 0.1, 0.2)	0.43	0.42



(a)



(b)

Fig. 6(a)Comparative analysis of Efficiency for S_1=(200,2,5) with Random Waypoint Mobility, and (b)Comparative analysis of Fairness for S_1=(200,2,5) with Random Waypoint Mobility.

Figure 6 compares the performance of the proposed model and that of the simple WFQ and FIFO allocations, using the Random Waypoint Mobility Model over 206 allocation cases. While the improvement in Fairness can be up to 16% (for an allocation case of (0.1, 0.1, 0.1, 0.2, 0.1, 0.1, 0.1, 0.2)), the Efficiency can be improved by 46% for a different allocation of (0.1, 0.1, 0.1, 0.1, 0.1, 0.15, 0.15, 0.2).

6. Conclusion

The proposed solution leverages Federated Learning (FL) to train models on a dataset generated from ns-3 simulations involving 8 wireless nodes using different mobility models. The FL model enables dynamic and decentralized resource allocation while preserving node privacy. Weighted Fair Queuing (WFQ) ensures Fairness in resource distribution, while Efficiency is maximized by predicting optimal throughput under varying mobility scenarios. Future work includes scaling the model to larger, more heterogeneous networks and incorporating real-world data and advanced mobility patterns. Optimizing energy efficiency, enhancing security against adversarial attacks, and enabling real-time adaptive learning will further refine the proposed solution.

References

[1] Davi Militani et al., “A Machine Learning Model to Resource Allocation Service for Access Point on Wireless Network,” *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Split, Croatia, pp. 1-6, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[2] Jihong Park et al., “Wireless Network Intelligence at the Edge,” *Proceedings of the IEEE*, vol. 107, no. 11, pp. 2204-2239, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[3] Brendan McMahan, and Daniel Ramage, Federated Learning: Collaborative Machine Learning Without Centralized Training Data, Google Research Blog, 2017. [Online]. Available: https://research.google/blog/federated-learning-collaborative-machine-learning-without-centralized-training-data/

[4] Robert Chang, Mahdi Rahimi, and Vahab Pournaghshband, “Differentiated Service Queuing Disciplines in NS-3,” *The Seventh IARIA International Conference on Advances in System Simulation*, pp. 17-23, 2015. [Google Scholar] [Publisher Link]

[5] Sangeetha Abdu Jyothi et al., “Measuring and Understanding Throughput of Network Topologies,” *SC '16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, Salt Lake City, UT, USA, pp. 761-772, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[6] Rong Yu, and Peichun Li, “Toward Resource-Efficient Federated Learning in Mobile Edge Computing,” *IEEE Network*, vol. 35, no. 1, pp. 148-155, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[7] M. Ashour, and Tho Le-Ngoc, “Performance Analysis of Weighted Fair Queues with Variable Service Rates,” *International Conference on Digital Telecommunications (ICDT'06)*, Cote d'Azur, France, pp. 51-55, 2006. [CrossRef] [Google Scholar] [Publisher Link]

[8] Mark Eisen et al., “Communication-Control Co-Design in Wireless Edge Industrial Systems,” *2022 IEEE 18th International Conference on Factory Communication Systems (WFCS)*, Pavia, Italy, pp. 1-8, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[9] Prateesh Goyal et al., “Backpressure Flow Control,” *Proceedings of the 2019 Workshop on Buffer Sizing*, Palo Alto CA USA, pp. 1-3, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[10] Tobias Hoßfeld et al., “Definition of QoE fairness in Shared Systems,” *IEEE Communications Letters*, vol. 21, no. 1, pp. 184-187, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[11] Chenmeng Wang et al., “Computation Offloading and Resource Allocation in Wireless Cellular Networks with Mobile Edge Computing,” *IEEE Transactions on Wireless Communications*, vol. 16, no. 8, pp. 4924-4938, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[12] Wei Yang Bryan Lim et al., “Federated Learning in Mobile Edge Networks: A Comprehensive Survey,” *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031-2063, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[13] Noor Zaman et al., “Energy Efficient Routing Protocol for Wireless Sensor Network,” *16th International Conference on Advanced Communication Technology*, Pyeongchang, Korea (South), pp. 808-814, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[14] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-Efficient Communication Protocol for Wireless Microsensor Networks,” *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, Maui, HI, USA, vol. 2, pp. 1-10, 2000. [CrossRef] [Google Scholar [Publisher Link]

[15] Natalie Temene et al., “A Survey on Mobility in Wireless Sensor Networks,” *Ad Hoc Networks*, vol. 125, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[16] Jingjing Yan, Mengchu Zhou, and Zhijun Ding, “Recent Advances in Energy-Efficient Routing Protocols for Wireless Sensor Networks: A Review,” *IEEE Access*, vol. 4, pp. 5673-5686, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[17] Yi-Han Xu et al., “Reinforcement Learning (RL)-Based Energy Efficient Resource Allocation for Energy Harvesting-Powered Wireless Body Area Network,” *Sensors*, vol. 20, no. 1, pp. 1-22, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[18] Padmalaya Nayak et al., “Routing in Wireless Sensor Networks using Machine Learning Techniques: Challenges and Opportunities,” *Measurement*, vol. 178, pp. 1-15, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[19] Enrique Tomás Martínez Beltrán et al., “Decentralized Federated Learning: Fundamentals, State of the Art, Frameworks, Trends, and Challenges,” *IEEE Communications Surveys & Tutorials*, vol. 25, no. 4, pp. 2983-3013, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[20] Yunfan Ye et al., “Edgefed: Optimized Federated Learning based on Edge Computing,” *IEEE Access*, vol. 8, pp. 209191-209198, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[21] Zhaohui Yang et al., “Delay Minimization for Federated Learning over Wireless Communication Networks,” *arXiv Preprint*, pp. 1-7, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[22] The ns-3 Network Simulator, 2015. [Online]. Available: http://www.nsnam.org

[23] S. Vimal et al., “Enhanced Resource Allocation in Mobile Edge Computing using Reinforcement Learning based MOACO Algorithm for IIoT,” *Computer Communications*, vol. 151, pp. 355-364, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[24] Diederik P. Kingma, and Jimmy Ba, “Adam: A Method for Stochastic Optimization,” *arXiv Preprint*, pp. 1-15, 2014. [CrossRef] [Google Scholar] [Publisher Link]