A Multilayer Perceptron Model for Station Grouping in IEEE 802.11ah Networks

Guan-Sheng Wang*, Chih-Yu Lin†, Yu-Chee Tseng*, and Lan-Da Van*
*Department of Computer Science, National Yang Ming Chiao Tung University, Taiwan
†Department of Computer Science and Engineering, National Taiwan Ocean University, Taiwan

NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium | 978-1-6654-7716-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/NOMS56928.2023.10154425

***Abstract*—With the rapid development of smart devices and wireless communication technologies, IEEE 802.11ah (WiFi HaLow) is designed to solve one of the major problems of Internet of Things (IoT): high collision probability in dense networks. It proposes the Restricted Access Window (RAW) mechanism, where stations (sensors) are partitioned into groups for time-division channel access. The grouping strategy, which highly influences network performance, needs to consider factors including the number of stations per group, and stations' data rates, and locations. With the advance of artificial intelligence technologies, we ponder whether deep learning can help solving this station grouping problem. In this paper, we propose a multilayer perceptron (MLP) model to predict RAW performance. More precisely, the model predicts the corresponding throughputs and packet loss rates of a given set of RAW configurations. Thus, based on the predicted results, we can determine proper RAW parameters. We have validated the proposed method by ns-3 simulations.**

***Index Terms*—IEEE 802.11ah, restricted access window (RAW), station grouping, multilayer perceptron, supervised learning**

I. Introduction

Internet of Things (IoT) has attracted great attention recently [1] and has many applications [2] [3] [4] [5]. With a massive number of stations transmitting data simultaneously, it is a challenge to legacy 802.11 protocols to meet access requirements [6].

IEEE 802.11ah, also known as Wi-Fi HaLow, is a wireless networking protocol published in 2017. It is a Wi-Fi standard for license-exempt 900 MHz band communications, which can support 1 Km long-range data transfer. Usually, star topology (i.e., all stations are connected to the base station directly.) is adopted in an IEEE 802.11ah network. Thus, stations may suffer severe collision when the number of stations becomes larger. IEEE 802.11ah aims to solve the major challenges of IoT, namely massive connectivity [7]. To address the challenges, IEEE 802.11ah proposes several new MAC features, such as Target Wake Time (TWT) and Restricted Access Window (RAW). TWT is a function that enables devices to determine when and how frequently they will wake up to access transmission medium. RAW is designed to reduce collisions among stations by station grouping [8]. In brief, it partitions stations into groups and allows one group to access channel at a time. It is similar to time division multiple access (TDMA), but has a soft RAW boundary.

In this work, our main focus is to find a good RAW scheduling strategy. Although several Wi-Fi HaLow hardware devices are available on the market, most performance studies on IEEE 802.11ah are based on mathematical models [9] [10], which may not actually reflect real network conditions due to their simplified or unrealistic assumptions. On the other hand, some network simulator 3 (ns-3) modules have been developed [11] to reflect real network scenarios. A proper RAW grouping policy can significantly impact network performance. However, IEEE 802.11ah standard does not specify its grouping policy [12].

The advance of machine learning motivates us to study whether AI techniques can be applied to the grouping problem in IEEE 802.11ah networks. In the proposed solution, we consider using ns-3 as a tool, which can iteratively run to generate a RAW performance dataset that can be used to train a multi-layer perceptron (MLP) model. The ns-3 tool runs by taking a group of IEEE 802.11ah stations' traffic parameters and a given RAW configuration to produce simulated performance indices of these stations within a beacon interval. As the dataset is sufficiently large, we can train a MLP as a predictor that maps a given group of stations' traffic parameters to a set of RAW configurations' predicted performance indices. Using these performance indices, one can pick a proper RAW configuration for use in a beacon interval. We have validated our proposal by extensive experiments.

II. Related Work

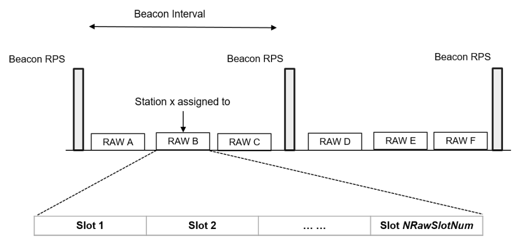


Fig. 1. The IEEE 802.11ah RAW scheme.

In IEEE 802.11ah, a RAW cycle may comprise multiple beacon intervals. In Fig. 1, the RAW cycle has two beacon intervals. Each beacon interval may open up one of more RAW groups. At the beginning of each beacon interval, an Access Point (AP) broadcasts RAW Parameter Set (RPS) information elements. For each RAW group to be opened up in a beacon interval, the RPS contains: start and end association IDs (AIDs), group start time, and duration. The other intervals excluding those occupied by RAW groups can be shared by all stations [10].

Several works have conducted evaluations on RAW performance. Reference [13] assesses the influence of the number of stations, traffic load and traffic distribution on the number of RAW groups and their durations. Nevertheless, the work does not provide an adaptive grouping strategy. Reference [9] shows that the RAW mechanism can improve fairness among groups. However, its grouping strategy does not follow the AID specification and is thus hard to realize. Reference [14] uses a surrogate model to determine network conditions by considering energy consumption and throughput. However, factors such as the distances to AP are not considered.

Recently, supervised learning has been studied in many disciplines. Machine learning techniques have also been applied to 4G LTE, and 5G networks. Reference [15] surveys the potentials, formulations, limitations, and future directions of machine learning for networks. A deep neural network for network traffic classification is proposed in [16]. A novel parameter searching method with Bayesian optimization configuration extrapolation is proposed in [17]. Reference [18] presents an Hidden-Markov Model that is capable of predicting throughput.

On the other hand, ns-3, a discrete-event network simulator, has been widely used for studying network performance. We adopt the IEEE 802.11ah module developed by [11], which has a RAW configuration interface for automatically and dynamically adjusting group configurations during simulation.

III. Supervised RAW Grouping

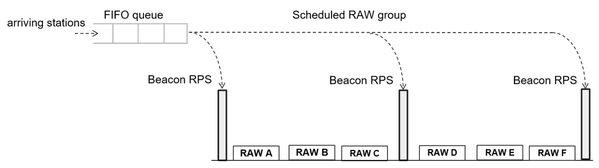


Fig. 2. Our RAW group assignment flow.

We consider an IEEE 802.11ah AP that is mainly designed to serve IoT devices. The basic unit of our scheduling is a beacon interval. We assume that all stations' network traffic parameters, including their sampling rates, packet sizes, and distances to the AP are already known. These parameters may be passively monitored by the AP or actively reported by stations. Our algorithm can handle various network parameters (e.g., various number of stations). In the beginning of each beacon interval, the AP will choose a number of stations that are allowed to transmit in the current beacon interval and determine their RAW configuration parameters. The goal is to assign these stations to RAW groups that may lead to higher throughput and lower packet loss rate. The beacon interval based scheduling is shown in Fig. 2.

The system architecture of the proposed method is shown in Fig. 3. There are three main components: ns-3 tool, model training, and model execution. The ns-3 tool generates a RAW performance dataset. As the dataset is sufficiently large, it can be used to train a MLP model as a predictor that can predict a set of RAW configurations' performances by given a group of stations' traffic parameters. Using these performance indices, we can pick a proper RAW configuration for use in an upcoming beacon interval.

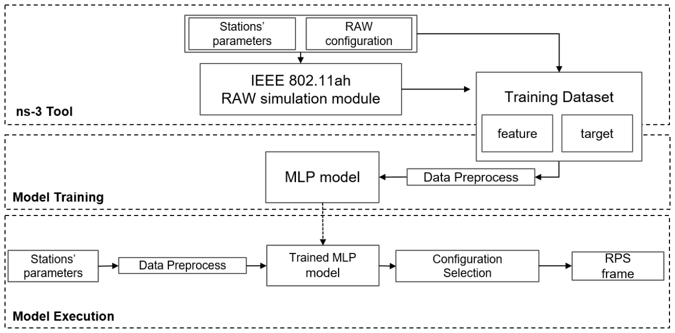


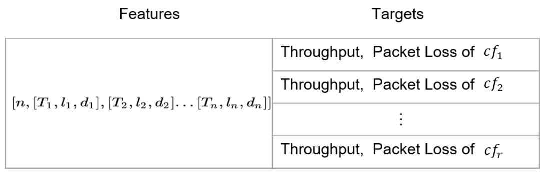
Fig. 3. System architecture.

A. ns-3 Tool

The IEEE 802.11ah RAW module developed by [11] has a configuration interface for automatically and dynamically adjusting group configurations during simulation. Based on this tool, we collect a training dataset for IEEE 802.11ah RAW grouping. Below is our simulation steps:

1) We generate a traffic parameter vector $[n, [T_1, l_1, d_1], [T_2, l_2, d_2], ..., [T_n, l_n, d_n]]$, where n is the number of stations and $[T_i, l_i, d_i]$ is the i_{th} station's average transmission interval, average packet length, and the distance to the AP, $i = 1...n$. Also, we assume that there are r configurations, denoted by $\{cf_1, cf_2, ..., cf_r\}$, under consideration. Each configuration $cf_i,\ i = 1...r$ is specified by three grouping parameters: $RAWslot$, $NRawSlotCount$ and $GroupingNumber$. $RAWslot$ indicates the number of RAW slots in a RAW group. $NRawSlotCount$ is used to calculate the length of a RAW slot:
$$\begin{aligned} RAWSlotDuration = 500us+ \\ NRawSlotCount \times 120us \end{aligned} \tag{1}$$
So the length of a RAW group is $RAWslot * RAWSlotDuration$. $GroupingNumber$ is the number of groups within the beacon interval. Here we use this number to evenly partition stations into groups. Note in Sec. III-C, when a configuration is selected for use, we will use these parameters to compute the RPS parameters.
2) ns-3 generates n virtual stations and an AP based on the above network parameters, and starts a round of ns-3 simulation by selecting a configuration in $\{cf_1, cf_2, ..., cf_r\}$. Each experiment will be conducted in UDP echo mode, lasting for a period of time.
3) At the end of a simulation round, the ns-3 simulator retrieves the throughput and packet loss of the configuration.

4) We repeat steps 2 and 3 for each configuration $cf_i, i = 1...r$. The simulation results are represented as $< feature, target >$ pairs as shown in Fig. 4.



Features	Targets
$[n, [T_1, l_1, d_1], [T_2, l_2, d_2] \ldots [T_n, l_n, d_n]]$	Throughput, Packet Loss of cf_1
	Throughput, Packet Loss of cf_2
	$\vdots$
	Throughput, Packet Loss of cf_r

Fig. 4. Training data format.

5) Repeat the above steps 1 to 4 with other station transmission parameters.

The above steps can be continuously run to collect training data. The detailed simulation parameters will be shown in Sec. IV.

B. Model Training

Next, we present how to train our MLP model. Note that the raw data in our simulation dataset needs to go through a preprocess before it can be used for model training. Recall that each feature has the format $[n, [T_1, l_1, d_1], [T_2, l_2, d_2], ..., [T_n, l_n, d_n]]$. However, in practice, the numbers of stations may vary. To train a MLP model, the input data size should be fixed. Therefore, we have to resize each data to a fixed length while not losing important features.

First, each raw data $[T_i, l_i, d_i], i = 1...n$, is converted to a vector $S_i = [\lambda_i, L_i, D_i, P_i]$ as follows:

$$\lambda_i = \frac{1}{T_i} \tag{2}$$

$$L_i = l_i * 0.01 \tag{3}$$

$$D_i = l_i * \lambda_i \tag{4}$$

$$P_i = log10\,(\frac{d_i \pi f}{c}) \tag{5}$$

Here, P_i is to represent the propagation error feature [11], where f and c are carrier frequency and the speed of light, respectively. Note that the preprocess process is designed based on the experiment results. For example, we found that the results by using the frequency is better than using the period. We also found that l_i should be rescaled so that we can get better results.

Next, we have to cluster these n vectors $S_1, S_2, ..., S_n$ into N clusters, where N is a fixed constant. The process is as follows. The first n/N vectors form the first cluster, the second n/N vectors form the second cluster, and so on. For each cluster, we calculate the standard deviation, median, and average of each sub-feature, That is, for $i = 1...N$, we compute $C_i = [[\lambda_{std}, \lambda_{med}, \lambda_{avg}], [L_{std}, L_{med}, L_{avg}], [D_{std}, D_{med}, D_{avg}], [P_{std}, P_{med}, P_{avg}]]$. Finally, we concatenate these C_is and append n at the end to form a new feature vector $[C_1, C_2, ..., C_N, n]$.

The MLP model is trained by the preprocessed dataset. The input feature is $[C_1, C_2, ..., C_N, n]$ and the outputs are two vectors $[th_1, th_2, ..., th_r]$ and $[pl_1, pl_2, ..., pl_r]$, where th_i and pl_i are the expected throughput and packet loss rate, respectively, of cf_i, $i = 1...r$.

C. Model Execution

As pointed out earlier, our scheduling is by beacon intervals. At the beginning of each beacon interval, our scheduler will dequeue a number of stations from the queue. To estimate how many stations to be dequeued, we use $\frac{(BI/T_i)*l_i}{dur(d_i)}$ as a measurement of the transmission time required by station i during a beacon interval BI, where $dur(d_i)$ is time required per bit. The total transmission time should not exceed one BI.

The model execution works as follows. First, a number of stations are dequeued. Their transmission parameters are preprocessed and then sent to our MLP model to predict the throughputs and packet loss rates of all configurations.

Then, we choose k configurations with the lowest packet loss rates as candidates. Then the one with the highest throughput is chosen. By this selection policy, we keep packet loss low, while maintaining throughput high.

After choosing the best configuration, we have to map it to the RPS information elements. Fig. 5 shows the RPS structure. There are six fields for each RAW group: slot definition format indication, cross slot boundary, slot duration count, number of slots, AID start and AID end. We set the first two fields to 1. The third and the fourth fields are filled by $NRawSlotCount$ and $RAWslot$ of the configuration. In the last two fields, we need to define a group's start AID and end AID. Since our strategy is to evenly partition stations into $GroupingNumber$ groups, the first RAW group's AID should start from 1 and end at $n/GroupingNumber$, the second RAW group's AID should start from $1+n/GroupingNumber$ and end at $2n/GroupingNumber$, and so on (we omit floor and ceiling for simplicity).

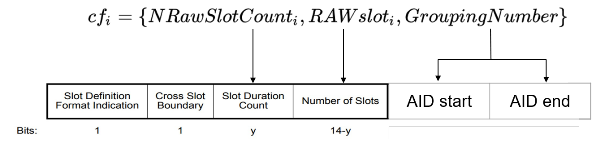

$cf_i = \{NRawSlotCount_i, RAWslot_i, GroupingNumber\}$
Slot Definition Format Indication
Cross Slot Boundary
Slot Duration Count
Number of Slots
AID start
AID end
Bits:
1
1
y
14-y

Fig. 5. RPS structure.

IV. Performance Evaluation

A. Parameters Selection

Table I lists station-related parameters. For each station, its traffic is determined by transmission interval (T), packet length (l), and distance to the AP (d). We sample these values between the minimum and the maximum values with a uniform distribution. The number of stations, n, is set between 80 and 125. We are unable to test more stations due to the limitation of the ns-3 tool. Note that the parameters used for training and testing are complete different so that we can

TABLE I
STATION TRAFFIC PARAMETERS

parameters	min	max
transmission interval(ms)	3000	10000
packet length $(bytes)$	90	110
distance to the access point (m)	20	225
n	80	125

TABLE II
RAW CONFIGURATION PARAMETERS

parameters	values
$GroupingNumber$	$\{3, 5, 8\}$
$NRawSlotCount$	$\{50, 100, 200\}$
$NRawSlotNum$	$\{1, 4, 7\}$

validate whether the trained model can be used for general cases.

Table II lists the RAW configuration parameters in our simulation. According to Eq. (1), we can calculate all RAW groups' durations, and their total should not exceed one beacon interval. Thus, our $NRawSlotCount$ may not exceed 200. According to the standard, the length of $NRawSlotNum$ is 3 bits. Thus, its value is bounded between 1 and 7. We adopt $\{1, 4, 7\}$ here. Based on these parameters, there are $r = 27$ configurations for RAW grouping in our simulation.

B. Performance Validation

Given station parameters, our goal is to find the best RAW configuration. Fig. 6 shows the validation results of our throughput and packet loss models, respectively. We compare ground truth (the results from ns-3) against the predicted values (the results from MLP model). While the predicted results for packet loss match well to the ground truth, there is some bias between the predicted throughputs and the ground truth. Fortunately, the relative trends are quite similar. Therefore, it is still possible to choose a proper RAW configuration from our prediction results. For example, from Fig. 6, our policy will pick cf_3.

To evaluate the performance of our model, we randomly pick 20 station configurations (i.e., varying the vector $\{n, [T_1, l_1, d_1], [T_2, l_2, d_2], ..., [T_n, l_n, d_n]\}$. For each station configuration, we run all 27 RAW configurations by simulations to obtain their throughputs and packet loss rates. Fig. 7 and Fig. 8 compare the ground truth maximum (gt max), median (gt median), and our predicted one for both performance indices. The results indicate that our MLP model can find a configuration that is very close to the one with the best throughput and is reasonably close to the one with the lowest packet loss rate. Note that since throughput and packet loss are conflicting factors, it is not always possible to find a configuration that meets both indices.

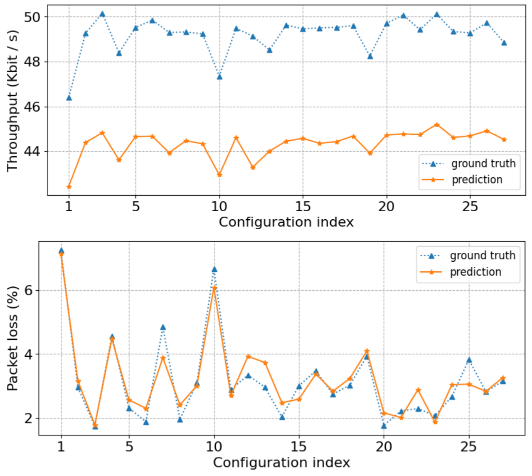


Fig. 6. Predicted throughput and packet loss rate vs. ground truth ($n = 83$, transmission interval∈[5000,9000]).

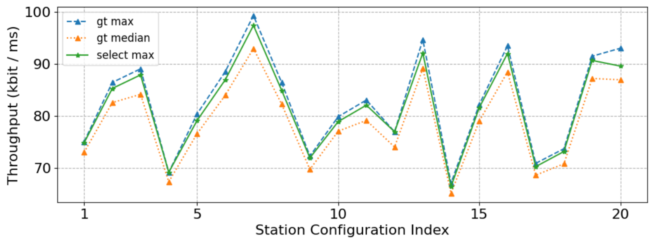


Fig. 7. Selection of configurations for throughput.

V. CONCLUSIONS

This paper presents a machine learning framework for studying 802.11ah RAW grouping by leveraging the ns-3 simulation tool. A data preprocessing method for transferring variable input data sizes to a fixed training data size is proposed that allows us to train two MLP models to predict RAW configurations' throughputs and packet loss rates. With the help of the MLP models, the framework allows one to select a proper RAW configuration in a per-beacon interval basis for dense IoT connection scenarios. For future work, we would continue to improve our work by more complicated models and more flexible configurations. For example, the RAW duration is assumed to be the same for all RAWs, which can be relaxed.

ACKNOWLEDGMENT

This work is supported by National Science and Technology Council, R.O.C under 110-2221-E-019-040-MY3.

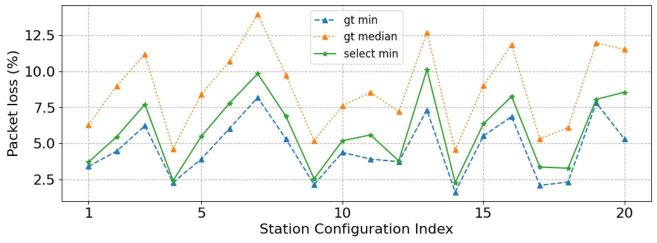


Fig. 8. Selection of configurations for packet loss rate.

References

[1] S. K. Sharma and X. Wang, "Toward Massive Machine Type Communications in Ultra-Dense Cellular IoT Networks: Current Issues and Machine Learning-Assisted Solutions," *IEEE Communications Surveys Tutorials*, vol. 22, no. 1, pp. 426–471, 2020.

[2] Y.-C. Wang, Y.-Y. Hsieh, and Y.-C. Tseng, "Multiresolution spatial and temporal coding in a wireless sensor network for long-term monitoring applications," *IEEE Trans. on Computers*, vol. 58, no. 6, pp. 827–838, 2009.

[3] M.-S. Pan, H.-W. Fang, Y.-C. Liu, and Y.-C. Tseng, "Address assignment and routing schemes for zigbee-based long-thin wireless sensor networks," in *IEEE VTC Spring*, 2008, pp. 173–177.

[4] C.-F. Huang, Y.-C. Tseng, and L.-C. Lo, "The coverage problem in three-dimensional wireless sensor networks," *Journal of Interconnection Networks*, vol. 08, no. 03, pp. 209–227, 2007.

[5] C.-H. Wu and Y.-C. Tseng, "Data compression by temporal and spatial correlations in a body-area sensor network: A case study in pilates motion recognition," *IEEE Transactions on Mobile Computing*, vol. 10, no. 10, pp. 1459–1472, 2011.

[6] Y. Zhou, H. Wang, S. Zheng, and Z. Lei, "Advances in IEEE 802.11ah standardization for machine-type communications in sub-1GHz WLAN," 06 2013, pp. 1269–1273.

[7] T. Adame, A. Bel, B. Bellalta, J. Barcelo, and M. Oliver, "IEEE 802.11ah: the WiFi approach for M2M communications," *IEEE Wireless Communications*, vol. 21, no. 6, pp. 144–152, 2014.

[8] E. Khorov, A. Lyakhov, A. Krotov, and A. Guschin, "A survey on IEEE 802.11ah: An enabling networking technology for smart cities," *Computer Communications*, vol. 58, pp. 53–69, 2015.

[9] T.-C. Chang, C.-H. Lin, K. C.-J. Lin, and W.-T. Chen, "Traffic-Aware Sensor Grouping for IEEE 802.11ah Networks: Regression Based Analysis and Design," *IEEE Transactions on Mobile Computing*, vol. 18, no. 3, pp. 674–687, 2019.

[10] L. Tian, E. Khorov, S. Latré, and J. Famaey, "Real-Time Station Grouping under Dynamic Traffic for IEEE 802.11ah," *Sensors*, vol. 17, no. 7, 2017.

[11] L. Tian, A. Seferagic, S. Santi, E. De Poorter, J. Hoebeke, and J. Famaey, "Extension of the IEEE 802.11ah ns-3 simulation module," 2018.

[12] *IEEE Std 802.11ah-2016 (Amendment to IEEE Std 802.11-2016, as amended by IEEE Std 802.11ai-2016)*, pp. 1–594, 2017.

[13] L. Tian, J. Famaey, and S. Latré, "Evaluation of the IEEE 802.11ah Restricted Access Window mechanism for dense IoT networks," in *2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2016, pp. 1–9.

[14] L. Tian, M. T. Mehari, S. Santi, S. Latré, E. De Poorter, and J. Famaey, "Multi-objective surrogate modeling for real-time energy-efficient station grouping in IEEE 802.11ah," *Pervasive and Mobile Computing*, vol. 57, pp. 33–48, 2019.

[15] M. Wang, Y. Cui, X. Wang, S. Xiao, and J. Jiang, "Machine Learning for Networking: Workflow, Advances and Opportunities," *IEEE Network*, vol. 32, no. 2, pp. 92–99, 2018.

[16] A. K. J. Michael, E. Valla, N. S. Neggatu, and A. W. Moore, "Network traffic classification via neural networks," University of Cambridge, Computer Laboratory, Tech. Rep. UCAM-CL-TR-912, Sep. 2017.

[17] O. Alipourfard, H. H. Liu, J. Chen, S. Venkataraman, M. Yu, and M. Zhang, "CherryPick: Adaptively Unearthing the Best Cloud Configurations for Big Data Analytics," in *USENIX Symposium on Networked Systems Design and Implementation*, 2017, pp. 469–482.

[18] Y. Sun, X. Yin, J. Jiang, V. Sekar, F. Lin, N. Wang, T. Liu, and B. Sinopoli, "CS2P: Improving Video Bitrate Selection and Adaptation with Data-Driven Throughput Prediction." ACM SIGCOMM, 2016.