Q-Learning based Collision Avoidance for 802.11 Stations with Maximum Requirements

Chang Kyu Lee[1], Dong Hyun Lee[2], Junseok Kim[2], Xiaoying Lei[3], and Seung Hyong Rhee[2*]
[1] Department of Network Business, Samsung Electronics Co., Ltd., Suwon, Korea
[e-mail: ck9104.lee@samsung.com]
[2] Department of Electronics Convergence Engineering, Kwangwoon University, Seoul, Korea
[e-mail: {selfldh1025, whdthsgnk, rhee}@kw.ac.kr]
[3] College of Information Engineering, Yangzhou University, Yangzhou, China
[e-mail: xylei@yzu.edu.cn]
[*]Corresponding author: Seung Hyong Rhee

Received September 1, 2022; revised December 20, 2022; accepted March 8, 2023; published March 31, 2023

Abstract

The IEEE 802.11 WLAN adopts a random backoff algorithm for its collision avoidance mechanism, and it is well known that the contention-based algorithm may suffer from performance degradation especially in congested networks. In this paper, we design an efficient backoff algorithm that utilizes a reinforcement learning method to determine optimal values of backoffs. The mobile nodes share a common contention window (CW) in our scheme, and using a Q-learning algorithm, they can avoid collisions by finding and implicitly reserving their optimal time slot(s). In addition, we introduce Frame Size Control (FSC) algorithm to minimize the possible degradation of aggregate throughput when the number of nodes exceeds the CW size. Our simulation shows that the proposed backoff algorithm with FSC method outperforms the 802.11 protocol regardless of the traffic conditions, and an analytical modeling proves that our mechanism has a unique operating point that is fair and stable.

Keywords: 802.11 WLAN, Q-learning, Backoff algorithm, Contention window, Resource allocation,

A preliminary version of this paper appeared in APIC-IST 2022, June 19-21, Gangneung, South Korea. This version includes a concrete modeling and analysis on the proposed resource allocation mechanism. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (2020R1F1A1069399), and in part by the Research Grant of Kwangwoon University in 2022.

http://doi.org/10.3837/tiis.2023.03.019 **ISSN : 1976-7277**

1. Introduction

In this paper, we discuss how to improve the collision avoidance algorithm applied to the CSMA/CA method of wireless LAN communication. The performance can be improved by applying reinforcement learning to the collision avoidance algorithm of the Binary Exponential Back-off (BEB) method, which is a collision avoidance algorithm between terminals used in the MAC protocol of Wireless Local Area Networks (W-LANs) [1]. A mobile node (MN) can transmit a frame after waiting for a backoff time that is selected randomly in the contention window (CW). In case a collision occurs, the size of CW is doubled to reduce the probability of collisions.

There have been many efforts to enhance the performance of the backoff algorithm in the WLAN MAC protocol: Serranoet al. [2] proposed an optimal size of contention windows to increase the overall network throughput. Also, Kang et al. [3] have suggested that the number of active nodes can be estimated by counting the number of idle slots and busy slots in a backoff period. Then, using the number of estimated nodes, each MN selects an optimal CW size. Ghazvini et al. [4] adaptively adjusted the contention window size by applying the game theory, i.e., via maximizing the payoff function of the network, each MN estimates the total number of nodes in the system and finds the minimum size of CW. Also, M.-H. Cheng et al. [5] discussed the performance of Linear Increase Linear Decrease (LILD) algorithm, in which the CW size is increased or reduced by one CWmin.

Recently, machine learning methods have been adopted to optimize the performance of the backoff mechanism in 802.11 MAC protocol. For example, Wydmański [6] uses a reinforcement learning method, in which the collision probability is used as the agent state information. Their model is proved to succeed in optimizing the contention window. Also, the methods for applying reinforcement learning [7] have been studied, and one of those method is applying Q-Learning using a frame with a time slot as a basic unit [8]. By applying this method, the data throughput can be maximized, and the delay time can be minimized. Y.-W. Chen [9] proposed a reinforcement learning method that is rewarded by throughput and optimally control the CW size periodically. Access point (AP) determines the CW size for the mobile nodes, and the nodes double the CW size after a collision (Mode 1) or keep the size selected by the AP (Mode 2).

In this paper, we design an efficient backoff algorithm that utilizes a reinforcement learning method to determine optimal values of backoffs. The mobile nodes share a common contention window (CW) in our scheme, and using a Q-learning algorithm, they can avoid collisions by finding and implicitly reserving their optimal time slot(s). In addition, we introduce Frame Size Control (FSC) algorithm to minimize the possible degradation of aggregate throughput when the number of nodes exceeds the CW size. We also provide an analytical model to show that our mechanism has a unique operating point that is fair and stable.

This paper is organized as follows: Section 2 presents our proposed Q-learning method for the multiple access control, and Section 3 describes an analytic model to study and control the fair resource allocation. Section 4 provides the results and discussion for the model proposed in the previous sections. After simulation results are shown and discussed, our analytical model is investigated study the stability and fairness property of the mechanism. Finally, Section 5 includes conclusions and suggestions on future works.

2. The Proposed Q-learning Method

2.1 Background

To avoid collisions, the IEEE 802.11 protocol uses BEB algorithm [1]. A node that has a frame to transmit, senses the carrier signal in the channel; if the channel is idle during DCF inter-frame space (DIFS) period, it starts the backoff timer before transmitting. The length of the timer is set to a randomly selected time slot in the contention window (CW)[0, W], where W is the current CW size. The selected backoff time is decreased by one after each time slot if the channel is idle, and the node transmits a data frame when the backoff time becomes 0. In **Fig. 1**, after a DIFS period, node 1 and node 2 randomly select 3 and 4, respectively, for their backoff timer. After 3 time slots, node 1 begins its transmission, and the backoff timer of node 2 stops decreasing until the end of next DIFS period, at which the timer starts again. If a transmission fails, the node doubles until $CW_{\max}$ is reached. Also, if a transmission succeeds or the maximum retransmission limit is reached, CW is initialized to $CW_{\min}$.

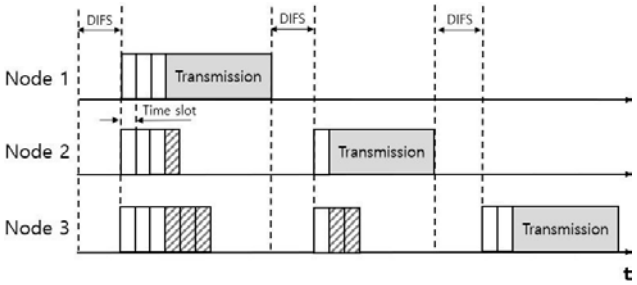


Fig. 1. Random backoff example of the IEEE 802.11 DCF:
Three nodes randomly select 3, 4, and 6, respectively, for their backoff timers

As CW size is reset to $CW_{\min}$ after a successful transmission, and is doubled after a failure; thus, as mentioned earlier, a node that succeeds a transmission is quite likely to gain the medium in the following contention. Also, if there are many nodes contending the channel, BEB algorithm suffers from a low efficiency and performance degradation.

Regarding the reinforcement learning, situations are mapped to actions to optimize a reward signal, i.e., the agent finds which action would give the maximum reward by trying those actions [7]. The agent continually interacts with the environment in order to select optimal actions, and the environment responds to the actions by presenting new states and rewards. The methods which estimate the action values and make action selections using the estimates, are collectively called action-value methods. The simplest rule to select a best action is to choose an action that has the highest estimated value, i.e., selected action is given by

$$A_t = \arg\max_a Q_t(a) \tag{1}$$

where $Q_t(a)$ is the action value at time t. Basically, the action value can be estimated by the average reward when that action is selected, and depending on system models, various forms of action value methods are possible. Also, while greedy selection of actions always exploits current knowledge, a balance between exploration and exploitation is always required in reinforcement learning. Probably, the most popular action value method in reinforcement learning is Q-learning, in which state-action values are learned using abehavior policy for the exploration [7]:

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)] \quad (2)$$

where $\alpha \in (0,1]$ is a step size parameter and $0 \le \gamma \le 1$ is a discount rate.

2.2 A Q-learning method for Collision Avoidance

In this section, we propose an efficient backoff algorithm that utilizes a reinforcement learning method to determine optimal values of backoffs in CW. The mobile nodes share a common contention window (CW) in our scheme, and using a Q-learning algorithm, they can avoid collisions by finding and implicitly reserving their optimal time slot(s). A node selects an optimal time slot to send its frame, and if the transmission succeeds, then the Q value of the slot will be increased and thus the nodes can reserve the time slot implicitly, which may greatly reduce the collision probability. All nodes are assumed to be synchronized with the fixed-size contention window.

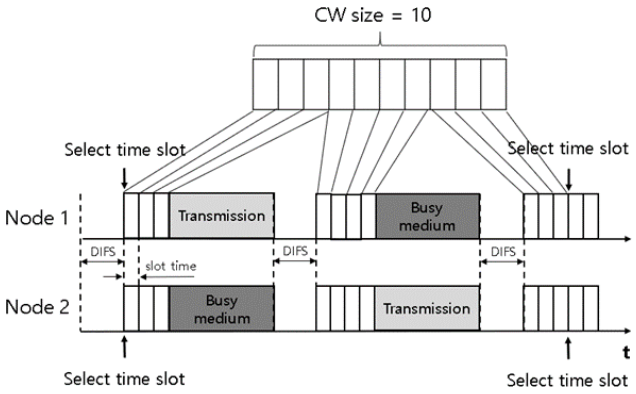


Fig. 2. Example of a frame with 10 time slots

Fig. 2 shows an example of a frame that has 10 time slots. Note that the length of a time slot is variable according to the atomic transmission sequence of frames. In the figure, if node 1 selects slot 3, then at the slot, the node starts the transmission sequence of (RTS-CTS)-Data-Ack frames. After the transmission and a free DIFS period, the time slot 3 ends and slot 4 begins. In the above example, node 2 selects slot 7 and it starts its transmission at the selected slot.

Algorithm 1 is the backoff algorithm for a single node. After the sequence of transmissions in the selected slot, the Q value of the slot is updated depending on whether the transmission succeeds or not. Through learning, each node adaptively finds the optimal value of backoff in the contention window. In this algorithm, the learning selection criterion uses the Upper Confidence Bound (UCB) [7] algorithm. If the number of exploitations for an action is not enough, then the upper bound could be larger and the action is quite likely to be selected. Using that mechanism, the node can efficiently learn the Q values by balancing exploration and exploitation.

Algorithm 1: Backoff algorithm with a fixed-size CW

```
Initially, Q values of all the slots are set to 0
while True do
    Wait for a beginning of a frame
    Select a slot that has a biggest Q value
    Wait for the selected slot
    Begin transmission
        RTS-CTS if needed
        Data-Ack
        Wait for a free DIFS
    Update Q value of the slot
```

Suppose that the total number of MNs in the system is less than the number of slots in a frame, and each node selects a single slot for their transmissions. Then some slots will not be used, and it may cause a performance degradation of the system. Thus, we also propose Frame Size Control (FSC) algorithm in the next section to avoid wasting time slots: If the total number of MNs in the system is less than the number of slots, then each node may occupy more slots. On the other hand, if the number of nodes is larger than the number of slots, then FSC algorithm increases the frame size to prevent a performance degradation due to collisions.

A Q-table example is given in **Fig. 3**, where CW size is 10. Using the action value method, mobile node 1 selects slot 3 at t_1. As the transmission of node 1 succeeds, the Q value of the time slot increases at t_2, the beginning of the next congestion window: With $\alpha = 0.1$, the Q value becomes 0.19. In the next contention window which begins at t_2, node 1 will choose the same slot again for its transmission, and the slot is implicitly reserved by the node.

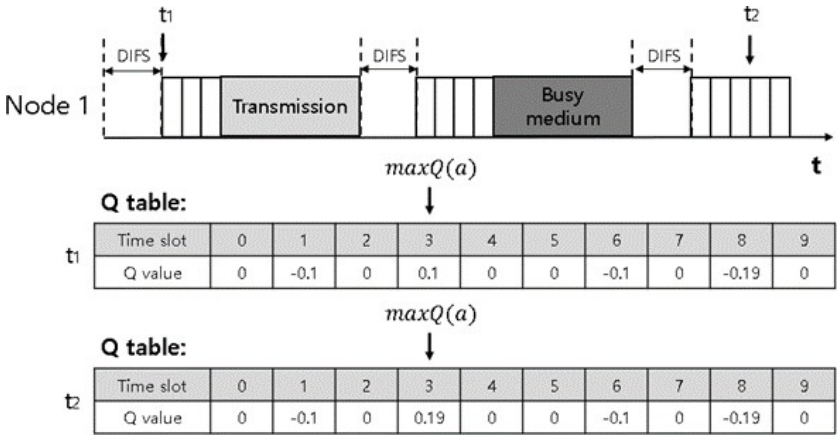


Fig. 3. Q value update after a transmission

3. Analytical Modeling and Control

3.1 Fair Resource Allocation

When transmitting data between a node and an AP, each node finds its own time slot through learning. As the number of slots in CW is fixed, if the number of nodes is less than the number

of slots, then each node may want to (implicitly) reserve as many slots as possible to increase their transmission throughputs. In this subsection, we propose a decentralized algorithm to allocate the slots fairly to the nodes.

Suppose that a set $K = \{1, \cdots, N\}$ of nodes is given and they have a common contention window of size W. Node $i \in K$ reserves several slots in the window via the learning process. We denote node i's number of (implicitly) reserved time slots by T_i, and the vector for all nodes in the system by $T = (T_1, \cdots, T_N)$. Then, vector T is said to be feasible if the following constraints are satisfied:

$$0 \le T_i \le M_i \ , \ \forall i \in K; \qquad T = \sum_{i \in \mathrm{K}} T_i \le W \tag{2}$$

where $0 < M_i \le W$ is the maximum requirement of node i. We assume that a node may have an upper bound on its own service demand, which is private to node i and its value may not be known to others. At the beginning of a congestion window, each node makes decisions on how many time slots it will use for transmission using the following algorithm:

A node first counts the number of time slots being used by all other nodes and then figures its fair share according to **Algorithm 2**. α_i is a real number set by node i, which determines the service quality, i.e., lower-class nodes have a lower value of α and a large α guarantees more time slots in the contention window. Thus if $\alpha_i = \alpha, \forall i$, all nodes in the network will have an equal number of slots within the window.

Algorithm 2: Computation of T_i

1: $T_{others} = \sum_{j \ne i} T_j$
2: $T_i = \alpha_i (W - T_{others}),\ 0 < \alpha_i < 1$
3: if $T_i > M_i$ then
4: $\quad T_i = M_i$
5: else if $T_i < 1$ then
6: $\quad T_i = 1$

Algorithm 2 is formulated from a distributed optimization problem as follows. We assume that each node has its own objective function given by (3) and tries to maximize the function value in a non-cooperative fashion.

$$J_i(\mathrm{T}) = T_i^{\beta i} \cdot ((W - T_{others}) - T_i) \tag{3}$$

where $0 < \beta_i \le 1$ is a weighting factor that is private to node i. As (3) is differentiable and concave, it is a specialform of the general objective function adopted in the networkutility maximization (NUM) problem [11]. It can be interpreted as benefit divided by cost, i.e., if a node implicitly reserves many time slots, then the node's performance will be high. On the other hand, if too many time slots are used, then the collision probability is increased due to the exploration of reinforcement learning process. One can see that the optimal policy T_i of node

i is given by **Algorithm 2**, where $\alpha_i = \frac{\beta_i}{\beta_i+1}$.

Also, note that the optimization problem of (2) can be implemented according to the gradient ascent algorithm as follows:

$$T_i^{t+1} = T_i^t + \lambda J'(T_i^t) \tag{4}$$

where λ is the learning rate. The value of T_i^0 starts with 1 andis repeatedly updated in the direction of steepest increase of J.

3.2 Frame Size Control

The frame size applied to Q-Learning is generally fixed initially. That is, the frame size is a value set by roughly predicting the number of MNs in the system. However, if the number of MNs accessing the AP at the same time exceeds the frame size, the AP becomes overloaded, and the performance decreases rapidly. To solve this AP saturation problem, we introduce the FSC algorithm that adjusts the frame size without interfering with Q-learning learning. **Algorithm 3** simply shows the operation of FSC.

When the AP is saturated, the frame size should be increased. Now when the AP is saturated, the optimal number of time slots at node i, T_i is exactly 1 due to the characteristics of the objective function. it is decided whether to operate the FSC. That is, when the AP becomes saturated, T_i is exactly 1, and FSC increase the frame size. In contrast, if it is not saturated, T_i greater than 1, and it is reduced again. This implementation is simple, but some coordination is needed to adjust the Q-value because the learning table of Q-Learning corresponds to a frame.

Algorithm 3: Frame Size Control

1: if $T_i \le M_i$ then
2: $W = W + 1$
3: else if $T_i > 1$ and $W > W_{init}$ then
4: $W = W - 1$

4. Results and Discussion

4.1 Simulation results

We have performed simulations in Python to evaluate the performance of our proposed algorithm. The nodes are assumed to have enough data frames to send all the time up to their personal demands. In **Table 1**, *CW size* is the sizeof the congestion window in **Algorithm 3**, $CW_{\min}$ and $CW_{\max}$ are parameters for the legacy DCF protocol.

Fig. 4 shows the simulation result of the backoff scheme proposed in this paper, and those of other algorithms for comparison. The performance is compared in terms of throughput versus the number of nodes. In our proposed algorithm, the learning rate and step size are both set to 0.001 and no upper bound is assumed in the nodes' service requirements. The overall throughput of 802.11 DCF is about 33 Mbps if there are 10 nodes in the network, and the throughput is decreased if there are more nodes in the system. The legacy DCF protocol suffers from the collisions due to the random backoff algorithm and it becomes severe as the node

numbers increases. LILD algorithm [5] shows a better performance from 20 nodes, and both modes of CW adjustment algorithm [9] outperform DCF and LILD. Note that our proposed scheme maintains a more throughput than CW adjustment algorithm regardless of the number of nodes. In the proposed algorithm, the nodes choose their best time slots by the reinforcement learning process such that almost all collisions can be avoided by the implicit reservation of timeslots.

Table 1. Simulation parameters

Parameter	Value
Number of nodes	10 - 50
CW size	100 slots
Data rate	54 Mbps
Data frame size	1500 bytes
DIFS	60 μs
Slot time	9 μs
CW_{min}	31
CW_{max}	1023

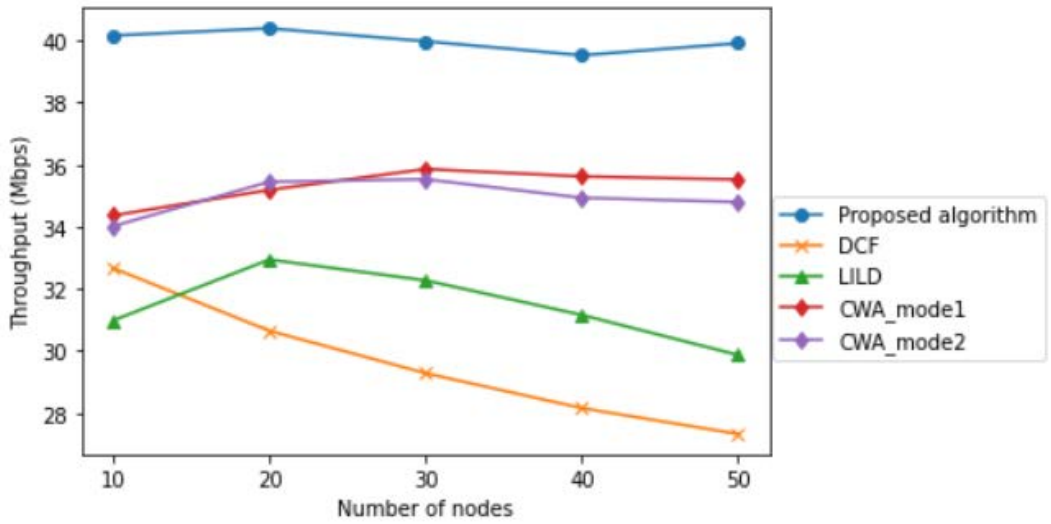


Fig. 4. Throughput of the backoff schemes: 802.11 DCF [1], LILD [5], CWA [9], and the proposed algorithm

Fig. 5 shows the performance improvement when Frame Size Control (FSC) algorithm is applied. We proposed FSC algorithm to minimize the possible degradation of aggregate throughput when the number of MNs exceeds that of time slots in the CW. One can see that, when FSC algorithm is adopted, the throughput is maintained regardless of the number of nodes.

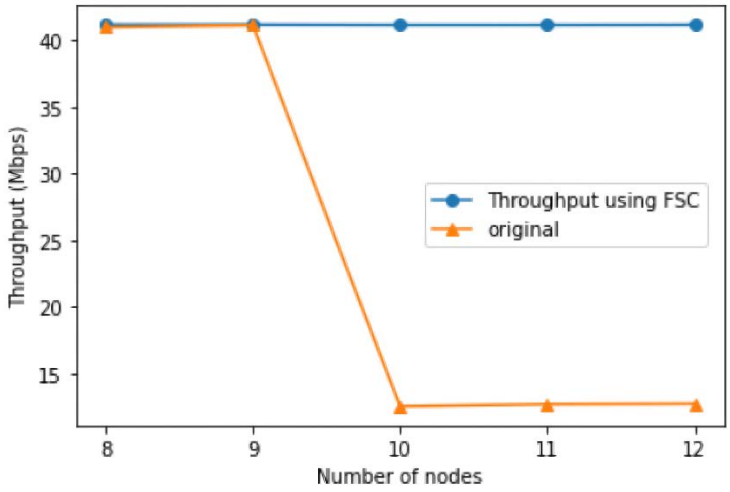


Fig. 5. The effect of FSC algorithm when the number of nodes exceeds the CW size.

4.2 Equilibrium of the Resource Allocation

As every node in K independently decides the number of time slots according to the objective function in (3), their optimization problems can be written as follows

$$\begin{aligned} &\max imize \quad J_i(\mathrm{T}), \quad \forall i \in \mathrm{K} \\ &subject\,to \quad 1 \le T_i \le M_i \end{aligned} \tag{5}$$

where we assume that every node $i \in \mathrm{K}$ has a value $0 < M_i \le W$, which is the maximum number of time slots which satisfies the service requirement of node i. Note that $J_i(\mathrm{T})$ is a real function that is strictly concave with respect to T_i and has its optimal at

$$\arg\max_{T_i} J_i(\mathrm{T}) = \alpha_i(W - T_{others}) \tag{6}$$

Each node in the system equation of (5) finds its optimal policy by (6), and slots in the contention window are selected via the reinforcement learning. It is a decentralized way of optimization, in which the mobile nodes non-cooperatively optimize their utility function. Such a model has been studied in many literatures. Note that the decentralized optimization model of Eq. (5) belongs to the resource allocation problem investigated in [12]. Using Theorem 2 in [12], which studies the Nash equilibrium point in a non-cooperative system, our system model of Eq. (5) has an equilibrium point that is unique. In addition, one can easily see that the resource allocation is fair, and the mobile nodes are satisfied with the allocation: I.e., no node can increase its utility function without decreasing other nodes' utilities.

Next, we study the convergence property of **Algorithm 3.** Let h be the time index which points the end of a time slot that has a successful state change: i.e., a time slot that is reserved bya node without a collision, or that is freed to be used. Also let $\mathrm{T}(h)$ be a K-dimensional vector such that component i denotes the number of time slots reserved for node i at time

h. Note that $\mathrm{T}(h)$ and $\mathrm{T}(h+1)$ is different in only one component of the vectors, A collision may occur in a time slot between h and $h+1$, and it causes the time slot is freedor another slot time is reserved at $h+1$. Then the system of (5) can be represented by the following iterations:

$$\mathrm{T}(h+1) = F(\mathrm{T}(h)), \quad h = 0,1,\ldots, \tag{7}$$

where F is a continuous mapping. As the nodes adaptively adjust their numbers of time slots, T needs to converge to the unique fixed point of (7), which is the operating point of the system.

Gauss-Seidel iteration method [13] is a well-known method for solving a system of linear equations: Only one node can update its value at a time and the nodes are assumed to receive the latest information on the system. In the iterative equation (7), $\mathrm{T}(h)$ and $\mathrm{T}(h+1)$ can be different only in their i^{th} element, and all the nodes obtains information on $\mathrm{T}(h)$ without any delay by carrier sensing. Thus, our implementation of the iterative algorithm corresponds to the Gauss-Seidel iteration method. One can implement Gauss-Seidel algorithms either in synchronous or asynchronous iterations. Note that, as the nodes in our system assumed to have no pre-specified order, they may join or leave the network anytime and even may not always active. Thus, the way of nodes' updates in our system model in (7) is asynchronous, and the nodes' implicit reservations will converge by the convergence property proved in [13]. Now, it is easily proved from the previous works for our system that the equilibrium point is unique, and the scheme always converges to the point. We provide the following result without proof.

Theorem 1. The operating point of the decentralized resource allocation problem in Eq. (5) is unique. In addition, if the asynchronous Gauss-Seidel method is adopted for the updates of the nodes' reservations, then the implicit reservation **T**(t) surely converges to the equilibrium point in a finite time.

4.3 Fair Share Property at the Equilibrium

In this section, we investigate the slot time reservation vector T^* determined at the Nash equilibrium. If we denote the total amount of reservations as $\mathrm{T}^* = \sum_i \mathrm{T}_i$, then node i's number of reserved slot times can be found by Theorem 2.

Theorem 2. For node $i \in \mathrm{K}$, its number of implicitly reserved time slots in the contention window is converged to the following value:

$$T_i^* = \begin{cases} M_i & , \quad if\ M_i < TS_i \\ 1 & , \quad if\ 1 > TS_i \\ TS_i & , \quad otherwise \end{cases}$$

where TS_i is a fixed point given by

$$TS_i = \frac{\alpha_i}{1-\alpha_i}(W - T^*) \tag{8}$$

Proof. According to Karuch-Kuhn-Tucker (KKT) conditions [14], we have the following constraints for any node i at the Nash equilibrium $\mathrm{T}^* = (T_1^*, \ldots, \mathrm{T}_\mathrm{K}^*)$

$$\lambda_i^* \geq 0, \mu_i^* \geq 0 \tag{9}$$

$$\lambda_i^* (T_i^* - M_i) = 0, \mu_i^* (1 - T_i^*) = 0 \tag{10}$$

$$-\alpha_i W_i + T_i^* + \lambda_i^* - \mu_i^* = 0 \tag{11}$$

where λ_i^* and μ_i^* are KKT multipliers, and $W_i = W - T_{others}$. One can deduce the following equalities from (9) and (10):

$$\lambda_i^* < \mu_i^* \quad \Rightarrow \quad \mu_i^* > 0 \Rightarrow T_i^* = 1$$
$$\lambda_i^* > \mu_i^* \quad \Rightarrow \quad \lambda_i^* > 0 \Rightarrow T_i^* = M_i$$

If we consider the fact that $W_i - T_i^* = W - T^*$, (11) can be written as

$$T_i^* - \frac{\alpha_i}{1-\alpha_i}(W - T^*) + \frac{1}{1-\alpha_i}(\lambda_i^* - \mu_i^*) = 0 \tag{12}$$

Now let $TS_i = \frac{\alpha_i}{1-\alpha_i}(W - T^*)$, then Eq. (12) and the above constraints give

$$M_i < TS_i \Rightarrow T_i^* < TS_i \Rightarrow \lambda_i^* > \mu_i^* \Rightarrow T_i^* = M_i$$
$$1 > TS_i \Rightarrow T_i^* > TS_i \Rightarrow \lambda_i^* < \mu_i^* \Rightarrow T_i^* = 1$$

If we remove the nodes in the first two cases and consider the unconstrained case, then the other case $T_i^* = FS_i$ can be proved easily.

Given α_i, W, and the nodes' requirements (if any), TS_i can be found as in Theorem 2. It is the fair amount of (implicit) reservation for a node and is also the solution of the recursion equation (11). Note that, according to the service requirement of a node, the amount of reservation can be different for the node. The following is a simple example of the formula in Theorem 2.

Example 3. **Fig. 6** shows the result of simulation with $W = 100$ and $\alpha_i = 0.5 (\beta_i = 1)$ for all i. At time 0 only one node exists in the system, and it uses the half of available time slots in CW according to Algorithm 2. As node 2 joins at 2 sec, node 1 decreases its usage and the (implicit) reservations of both nodes converge their fair shares as given in Theorem 2; i.e., $TS_i = (100 -_i TS_i) = 33$ for all i. At 4 sec, node 3 joins, and the node is assumed that its service requirement is bounded such that $M_3 = 16$. One can see that T_3 is quickly converges to its maximum requirement and other nodes also converge to their fair shares, which is $TS_i = (100 -_i TS_i) = 28$ for all $i \neq 3$.

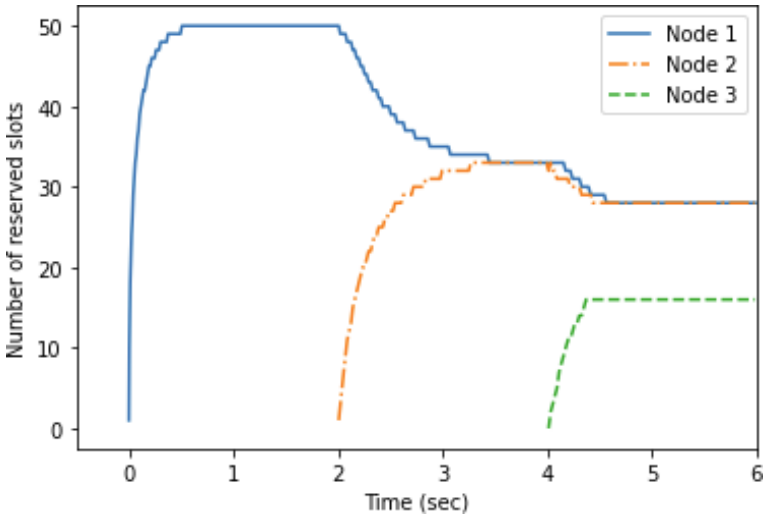


Fig. 6. Each node dynamically adjusts its use of time slots as the nodes join the network. Node 3's service requirement is bounded.

5. Conclusions

We proposed a new backoff algorithm that utilizes a reinforcement learning method to determine optimal values of backoffs. Also, we suggested Frame Size Control algorithm to minimize the possible degradation of aggregate throughput. Simulation results showed that our backoff method with the FSC algorithm outperforms the 802.11 protocol regardless of the traffic conditions. In addition, we studied the properties of the Nash equilibrium and its convergence property. Using the analytical modeling, we can compute the fair share allocation at the equilibrium. Several issues should be studied further: We may extend the proposed mechanism to a multi-hop network environment. Also, the convergence speed may be particularly important for a practical application, especially when many nodes are contending in the system.

References

[1] Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Standard 802.11, 2016. Article (CrossRef Link)

[2] P. Serrano, P. Patras, A. Mannocci et al., "Control theoretic optimization of 802.11 WLANs: Implementation and experimental evaluation," *Computer Networks*, vol. 57, no. 1, pp. 258–272, 2013. Article (CrossRef Link)

[3] S. Kang, J. Cha, and J. H. Kim, "A Novel Estimation-Based Backoff Algorithm in the IEEE 802.11 Based Wireless Network," in *Proc. of IEEE Consumer Communications and Networking Conference*, 2010. Article (CrossRef Link)

[4] M. Ghazvini, N. Movahedinia, and K. Jamshidi, "A game-theory based contention window adjustment for IEEE 802.11 under heavy load," *International Journal of Communication Networks and Information Security*, vol. 5, no. 2, pp. 93–103, 2013. Article (CrossRef Link)

[5] M.-H. Cheng, W.-S. Hwang et al., "A oneself adjusts backoff mechanism for channel access in IEEE 802.11 DCF WLAN," in *Proc. of Int. Conf. on Complex, Intelligence, and Software Intensive Systems*, 2013. Article (CrossRef Link)

[6] W. Wydmanski and S. Szott, "Contention window optimization in IEEE 802.11 ax networks with deep reinforcement learning," in *Proc. of IEEE WCNC*, 2021. Article (CrossRef Link)

[7] R. Sutton and A. Barto, *Reinforcement learning: An introduction*, MIT press, 2018. Available: http://incompleteideas.net/book/the-book-2nd.html

[8] C. K. Lee and S. H. Rhee, "Collision Avoidance in IEEE 802.11 DCF using a Reinforcement Learning Method," in *Proc. of ICTC*, Oct. 2020. Article (CrossRef Link)

[9] Y.-W. Chen and K.-C. Kao, "Study of contention window adjustment for CSMA/CA by using machine learning," in *Proc. of APNOMS*, Sep. 2021. Article (CrossRef Link)

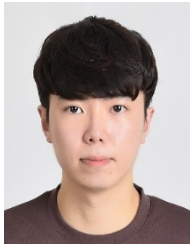
[10] R. Agrawal, M. Hedge, and D. Teneketzis, "Asymptotically efficient adaptive allocation rules for the multiarmed bandit problem with switching cost," *IEEE Transactions on Automatic Control*, vol. 33, no. 10, pp. 899–906, 1988. Article (CrossRef Link)

[11] D. Palomar and M. Chiang, "Alternative distributed algorithms for network utility maximization: Framework and applications," *IEEE Transactions on Automatic Control*, vol. 52, no. 12, pp. 2254–2269, 2007. Article (CrossRef Link)

[12] S. H. Rhee and T. Konstantopoulos, "A decentralized model for virtual path capacity allocation," *IEEE INFOCOM*, NY, USA, pp. 497–504, 1999. Article (CrossRef Link)

[13] D. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Athena Scientific, 2015.

[14] D. Bertsekas, *Nonlinear Programming*, Nashua, NH, Athena Scientific, 2016.



Chang Kyu Lee received the B.S. and M.S. degree from Dept. of Electronics Convergence Engineering of Kwangwoon University, Seoul, Korea, in 2018 and 2020, respectively. In 2021, he joined the Department of Network Business in Samsung Electronics Co. Ltd., Suwon, Korea. His research interests include developing solutions and platforms to ensure LTE/NR mobile communication network performance.

Dong Hyun Lee received his B.S. degree in 2018 and is expected to obtain his M.S. degree in 2023, both from the Dept. of Electronics Convergence Engineering of Kwangwoon University, Seoul, Korea. His research interests include power saving mechanisms in wireless networks and power control in the ad-hoc network using machine learning and deep learning methods. Mr. Lee is a member of KSII.

Junseok Kim received his B.S. degree in 2018 and is expected to obtain his M.S. degree in 2023, both from the Dept. of Electronics Convergence Engineering of Kwangwoon University, Seoul, Korea. His research interests include multiple access control in wireless networks using reinforcement learning methods. Mr. Kim is a member of KSII.

Xiaoying Lei received the B.S. degree from North China Electric Power University, Beijing, China in 2006, and the M.S. and Ph.D. degrees from Kwangwoon University, Seoul, Korea in 2010 and 2015, respectively. In 2015, she joined the College of Information Engineering at Yangzhou University, Yangzhou, Jiangsu, China. Her research interests include mmWave VANETs, machine learning, federated learning. Dr. Lei is a member of IEEE.

Seung Hyong Rhee received the B.S. and M.S. degrees from Yonsei University, Seoul, Korea, in 1988 and 1990, respectively, and the Ph.D. degree from the Electrical & Computer Engineering Department of the University of Texas at Austin, Texas, USA, in 1999. From 1990 to 1995, he was with Agency for Defense Development, Korea, working on defense communication networks. In 2000 he joined the Department of Wireless Communication Engineering at Kwangwoon University, Seoul, Korea. His research interests include optimal and decentralized flow control in high-performance networks, security architecture and protocols, and reinforcement learning approaches for the wired and wireless Internet. Dr. Rhee is a member of IEEE and ACM.