To Send or Not To Send - Learning MAC Contention

SaiDhiraj Amuru†, Yuanzhang Xiao†, Mihaela van der Schaar‡, R. Michael Buehrer†
†Wireless@VT, Department of Electrical and Computer Engineering, Virginia Tech
‡ Department of Electrical Engineering, UCLA
Email: {adhiraj, yxiao, rbuehrer}@vt.edu, mihaela@ee.ucla.edu

***Abstract*—The exponential back-off mechanism, proposed for reducing MAC-layer contention in the 802.11 standard, is sub-optimal in terms of the network throughput. This back-off mechanism and its improved variants are especially inefficient under unknown dynamics such as packet arrivals and user entry/exit. In this paper, we formulate the problem of optimizing this back-off mechanism as a Markov decision process, and propose online learning algorithms to learn the optimal back-off schemes under unknown dynamics. By exploiting the fact that some components of the system dynamics (such as protocol states) are known because the users follow the common 802.11 protocol, we propose a post-decision state (PDS)-based learning algorithm to speed up the learning process. Compared to traditional Q-learning algorithms, the advantages of the proposed online learning algorithm are that 1) it exploits partial information about the system so that less information needs to be learned in comparison to other learning algorithms, and 2) it removes the necessity for action exploration which usually impedes the learning process of conventional learning algorithms (such as Q-Learning). We prove the optimality of the proposed PDS-based learning algorithm and via numerical results demonstrate the improvement over existing protocols and Q-learning in terms of throughput and convergence speed. We first address this problem from a single-user perspective and later describe the challenges involved and present new insights into the multi-user learning scenarios, especially in cases where the MDP models of the users are coupled with each other.**

I. Introduction

Contention-based protocols are the de facto MAC-layer access schemes in modern day wireless networks [1]. They have been shown to outperform standard multiple access schemes such as TDMA and FDMA when the network load is low (few users in the network) and all users do not have packets to transmit at every time instant [1]. However, when the network load increases, random access schemes tend to create more collisions and as a result, pave the way for protocols based on scheduling [2]. This can be achieved only at the expense of a centralized controller which may not exist in most decentralized wireless networks. Hence, in this paper, we consider improving the performance of the distributed coordination function (DCF) protocol (that addresses contention-based medium access in 802.11) via reinforcement learning (RL) techniques.

DCF uses Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) to achieve multiple access communication. In addition, it can employ a 4-way handshaking mechanism based on the RTS (request to send)-CTS (clear to send) protocol to transmit data packets; see [1] for more details. Upon receiving an acknowledgement (ACK) for the transmitted data packets, the source node enters an idle/wait state before it starts transmitting again. In the wait state, it chooses a back-off window and a random back-off time (which is between 0 and the back-off window), and counts down the back-off time in the time instants when the medium is sensed to be idle (*i.e.*, no other node is using the channel). The node can again access the wireless medium when its back-off counter expires (*i.e.*, the back-off time is 0). After data transmission, if the ACK is received successfully, the node chooses the minimum back-off window and if not, it enters into an exponential back-off mode where the back-off window size is doubled for every retransmission, until a limit is reached.

Collisions can occur when different nodes in the network send an RTS at the same time (which may result in packet corruption) and thereby do not receive a CTS in response to their request. When this happens, each node doubles the previously chosen back-off window size and enters the wait state. Such an exponential back-off mechanism reduces the number of collisions for two reasons. First, there are fewer attempts to access the medium due to the back-off. Second, there are even fewer *simultaneous* attempts to access, because the back-off time is randomly chosen and is likely to be different across different nodes/users.

Various studies have shown that this exponential back-off mechanism is not efficient in terms of network utilization [3]-[13]. For example, the back-off window size chosen should be higher (lower) when the number of users *i.e.*, the network load increases (decreases) so as to avoid frequent collisions. Moreover, it is important not to reset the back-off window size to its minimum value after every successful transmission. This intuition led to several heuristic solutions [3]-[13]. These approaches choose the back-off window size based on the number of users present in the network (which is often impossible to estimate) or the priority of the users in the network. However, the previous works [3]-[13] do not give attention to the arrival process of data packets, which directly affects the transmission strategies. For example, when the number of packets in the buffer is higher, the user may choose smaller back-off values so as to transmit data packets more often and avoid any buffer overflows or holding costs. Also, prior works fail to provide theoretical guarantees about the performance of these heuristics. In this work, we theoretically prove that the proposed learning algorithms converge to the optimal strategies. Table I shows a comparison of the related works.

TABLE I. Comparison between related works

	Heuristic Algorithms [3]-[13]	Q-Learning	PDS-Learning (proposed)
Unknown Packet Arrivals Considered	No (model assumed)	Yes	Yes
User Entry/Exit	Yes	Yes	Yes
Optimal back-off	No	Yes (asymptotically)	Yes
Convergence time	Slow	Slow	Fast
Action exploration	N/A	Yes	Not required
Environment structure exploitation	No	No	Yes

In this paper, we study the MAC layer back-off window optimization problem from a RL perspective by developing a PDS-based learning algorithm and obtain the optimal transmission strategies (that choose the optimal back-off time) when the packet arrival process and network load are unknown *a priori*

and can be dynamic. The RTS-CTS handshake mechanism is modeled via a Markov decision process (MDP) approach in Section II. A key difference from the standard 802.11 protocol is that we choose the back-off window based on the system state optimally, instead of the exponential mechanism proposed in the 802.11 standard. A novel decomposition rule developed in Section III splits the MDP into known and unknown components to speed up the convergence of the RL algorithm. As will be shown in Section IV, the proposed algorithm performs better than the state-of-the-art algorithms [3]-[13]. We also discuss the behavior of these single-user learning algorithms when they are extended to multi-user scenarios in Section IV and explain the issues that arise when dealing with multi-user systems. Specifically, we consider situations where the users' *MDPs are coupled* with each other but they cannot communicate with others, which is different from the assumptions made in [14]-[16]. These issues were *not* discussed before in the literature and hence provide a new perspective on the commonly encountered multi-user scenarios. Finally, we conclude the paper in Section V.

II. MDP Model

We consider a wireless network with an unknown number of users communicating on a single frequency channel. We first study the MAC layer contention access problem from a single user perspective and treat all other users as part of the environment. Hence, we model the network load (*i.e.*, the transmissions from other users) via an unknown parameter ρ that indicates the probability with which the user under study faces collisions. For example, a higher (lower) value of ρ indicates the presence of a larger (smaller) number of users [2]. We next discuss the MDP model formulation of this 802.11 wireless network, which is also shown in Fig. 1.

A. Wireless Medium State

The wireless medium state at time n, denoted by m^n, can belong to one of the two possible states in the set $\mathcal{M} = \{\text{BUSY}, \text{FREE}\}$. The state is BUSY when the channel is being used by others, and is IDLE otherwise. Note that it is taken to be IDLE when the user under study is using the channel. Since the back-off window sizes chosen by other users are unknown, the evolution of the medium state from m^n to m^{n+1} is unknown. More specifically, the medium state transition probabilities are unknown. We assume that once the user accesses the wireless channel, the medium stays in the FREE state until the packet transmission is completed.

B. 802.11 Protocol States

To denote the various events involved in the RTS-CTS protocol, the protocol state at time n is denoted by pr^n which can take one of the following 4 possible values $pr^n \in \mathcal{P} = \{\text{SEND RTS}, \text{RTS COLLISION}, \text{CTS RECEIVED}, \text{WAIT}\}$. When the medium is sensed to be BUSY, the user is assumed to be in the WAIT state by default. When the medium is sensed to be FREE, the user decrements the back-off counter to 0 and then tries to reserve the channel by sending a RTS packet to its destination node. At this point, the protocol state is taken to be SEND RTS. Depending on whether this RTS packet collided with another users' RTS or not, the protocol enters the RTS COLLISION or CTS RECEIVED states with probability ρ and $1-\rho$ respectively. When the RTS faces collision (identified by the fact that the source node did not receive a CTS packet within a pre-specified time), the user enters the WAIT state and waits for its turn to access the channel again (collisions may occur in other states as well which are ignored for ease of exposition in this paper, but can be easily included into this model by considering additional state transitions). However, if no collision occurs, the protocol enters the CTS RECEIVED state where it sends a data packet to its intended destination node and awaits an ACK packet. It is assumed that the ACK packet is received without any errors via the feedback channel. Also, it is assumed that the user is aware of the probability (PLR = packet loss rate) of packet failure.

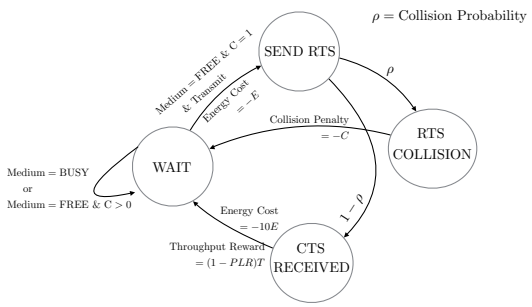

Fig. 1. MDP model of the user in a 802.11 network.

C. Back-off State

Note that in this work, as mentioned earlier, we choose the back-off window based on the system state (details follow shortly). Specifically, we choose it in the range $[CW_{\min}, CW_{\max}]$ where $CW_{\min}$ and $CW_{\max}$ are the minimum and the maximum values that can be chosen as per the 802.11 standard. Once a back-off window is chosen, a back-off counter keeps track of the back-off value as follows.

The user's transmission decision depends on the remaining back-off when the medium is sensed to be FREE. The remaining back-off at time n is denoted by $c^n \in [0, CW_{\max}]$ where $CW_{\max}$ is the maximum back-off that can be chosen by the user when it enters the WAIT state. When the medium state is BUSY, the back-off state is frozen *i.e.*, $c^{n+1} = c^n$. When the medium is FREE and the protocol is in WAIT state, the state evolves as $c^{n+1} = \max(0, c^n - 1)$. When the user is in the SEND RTS protocol state, the back-off state is again frozen *i.e.*, $c^{n+1} = c^n$. Since the user enters the WAIT state from the RTS COLLISION and CTS RECEIVED states, the user chooses a new back-off window size (this choice is discussed shortly) and this value is taken to be the next state c^{n+1}.

D. Transmission Buffer State

The transmission buffer follows the first-in first-out queuing principle [19]. At time n, l^n data packets are injected into the transmit buffer according to the distribution $p^l(l)$ which is unknown *a priori*. The transmit buffer can hold a maximum of B packets. Thus the buffer state $b \in \mathcal{B} = \{0, 1, 2, \ldots, B\}$ evolves as $b^{n+1} = \min(b^n - f^n + l^n, B)$, where f^n is the number of packets transmitted at time n. It is easy to see that f^n depends on a) whether the user got access to the wireless channel or not and b) whether the packet faced any collisions or incorrectly decoded due to bad wireless channel conditions. More specifically, $f^n \geq 1$ with probability $1 - PLR$ when m^n = FREE and pr^n = CTS RECEIVED; else it is 0 (as

the user does not transmit packets in any other states). As mentioned earlier, the unknown arrival rate and the buffer state evolution directly influences the back-off window sizes chosen.

Based on the above discussions, the overall state of the environment at time n is given by $s^n = [b^n, m^n, pr^n, c^n]$. Since the packet arrival rate and the network load are unknown, the buffer and the medium state transition probabilities are unknown *a priori*. Along similar lines, the probability with which the protocol transitions from SEND RTS state to RTS COLLISION or CTS RECEIVED states is unknown because it depends on ρ. In contrast, based on the discussion in Section II-C, the state transition probability of the back-off window is known given the protocol states. The known and unknown system dynamics, which will be used to speed up the learning process of PDS, will be discussed in more detail in Section III.

E. MDP Actions

The user must choose the best transmission probability in order to maximize its rewards in the presence of other users in the network. It does this by choosing the back-off window optimally. Specifically, the user can choose an action (the back-off value at this time instant is therefore the action chosen) denoted by $a^n = cw^n$ where $cw^n \in [CW_{\min}, CW_{\max}]$. The action choice depends not only on the estimated network load, but also on the buffer size b^n as will be discussed soon.

F. Rewards and Costs - Feedback for the MDP

The user gets feedback from the environment in terms of the rewards and costs associated with the various actions i.e., back-off windows chosen. They enable the user to modify its actions in an attempt to maximize its overall reward. In this work, since we only consider the MAC layer protocol, we restrict ourselves to the following rewards/costs - throughput obtained, energy expended and penalty due to RTS collision. When the user transmits a packet successfully in the CTS RECEIVED state, it receives a positive reward T. In this paper, we consider the expected throughput reward *i.e.,* $(1-PLR)T$ when the user enters the CTS RECEIVED state (as it sends data correctly with probability $1-PLR$). The user expends energy E when it sends a RTS packet and energy $10E$ when it sends a data packet (captures the effect of longer data packet in comparison to the control packet). If the protocol enters the RTS COLLISION state, then there is a penalty C as the user did not choose the appropriate back-off window to avoid collisions. The user maximizes the cumulative discounted reward over an infinite time horizon. We next discuss the post-decision state-based (PDS) learning algorithm which uses these rewards to learn the optimal actions.

III. PDS-BASED ONLINE LEARNING

Conventional learning algorithms such as Q-learning learn the value of the state-action pairs, in an attempt to obtain the optimal policies, under the assumption that the system dynamics are completely unknown *a priori*. Hence, the optimality of such learning algorithms is only guaranteed when all possible state-action pairs are visited infinitely many times. However, in practice, rewards/costs and the transition probabilities are only partially unknown *a priori*. The known dynamics of the environment structure can be exploited to develop efficient learning algorithms that are significantly faster than Q-learning. Post-decision state-based (PDS) learning [17]– [19] is one such algorithm that separates the known and the unknown components (both rewards/costs and the state transition probabilities), to achieve an improved and faster learning performance.

We briefly explain the PDS algorithm in the context of the 802.11 framework introduced in Section II. A post-decision state describes the environment after the known dynamics take place and before the unknown dynamics can occur. Denote the PDS as $\tilde{s} \in \mathcal{S}$ where $\mathcal{S}$ indicates the set of possible environment states described in Section II (*i.e.*, the set of PDSs is the same as the set of original environment states). For the problem under consideration, the PDS $\tilde{s}^n$ at time n is related to the states s^n and s^{n+1} as below;

PDS at time n: $\tilde{s}^n = [b^n - f^n, \tilde{m}^n, \tilde{p}^n, \tilde{c}^n]$,
State at time $n+1$: $s^{n+1} = [b^n - f^n + l^n, m^{n+1}, p^{n+1}, c^{n+1}]$.

While we only showed the evolution of the buffer state above, the states $\tilde{c}^n, \tilde{p}^n$ are known in some cases depending on the medium and the protocol states at time n. For example, when $m^n =$ FREE and $pr^n =$ WAIT, $c^{n+1} = \tilde{c}^n = \max(c^n - 1, 0)$ because the back-off value is decreased when the medium is free. See Fig. 1 for more details.

Using PDS, the state transition probability function can be factorized into the known and the unknown components as

$$p(s^{n+1}|s^n, a^n) = \sum_{\tilde{s}^n} p_k(\tilde{s}^n|s^n, a^n) p_u(s^{n+1}|\tilde{s}^n, a^n), \quad (1)$$

where p_k is the known state transition from s^n to PDS $\tilde{s}^n$ and p_u accounts for the unknown state transition from $\tilde{s}^n$ to s^{n+1}. The reward/cost function can be factored as

$$r(s^n, a^n) = r_k(s^n, a^n) + \sum_{\tilde{s}^n} p_k(\tilde{s}^n|s^n, a) r_u(\tilde{s}^n, a^n) \quad (2)$$

where $r(s^n, a^n)$ is the overall reward obtained in state s^n when action a^n is performed; $r_k(s^n, a^n)$ denotes the known rewards/costs and $r_u(\tilde{s}^n, a^n)$ indicates the unknown rewards/costs.

For the current problem, p_k is given by

$$p_k(\tilde{s}^n|s^n, a^n) = p^f(b^n - \tilde{b}^n|m^n, pr^n) \times p^c(c^{n+1}|c^n, pr^n, m^n, cw^n), \quad (3)$$

where $p^f(b^n - \tilde{b}^n|m^n, pr^n)$ is the probability of sending $b^n - \tilde{b}^n$ packets (it is known because the user sends packets only in the CTS RECEIVED state) and $p^c(c^{n+1}|c^n, m^n, pr^n, cw^n)$ is the probability that the back-off changes to c^{n+1} from c^n. The unknown transition probability p_u is given by

$$p_u(s^{n+1}|\tilde{s}^n, a^n) = p^l(b^{n+1} - \tilde{b}^n) \times p^m(m^{n+1}|m^n, pr^n) \times p^{protocol}(pr^{n+1}|pr^n, m^n, c^n), \quad (4)$$

where $l^n = b^{n+1} - \tilde{b}^n$ is the unknown number of packets entering the buffer at time n, $p^m(m^{n+1}|m^n, pr^n)$ indicates the medium state transition probability (depends on ρ), and $p^{protocol}(pr^{n+1}|pr^n, m^n, c^n)$ is the protocol state transition probability (independent of the action taken as the state variables m^n, pr^n and c^n decide the next state). p_u is independent of the actions a^n, a feature that can be exploited to increase the convergence rate of PDS, which will be discussed shortly.

All the rewards/costs are known in the current model, therefore $r_u(\tilde{s}^n, a^n) = 0$.

Following [19], the optimal PDS value function $\tilde{V}^*(\tilde{s}^n)$ is

$$\tilde{V}^*(\tilde{s}^n) = \sum_{s^{n+1}} p_u(s^{n+1}|s^n) V^*(s^{n+1}) \quad (5)$$

$$V^*(s^n) = \max_{a^n \in \mathcal{A}} \Big\{ r_k(s^n, a^n) + \gamma \sum_{\tilde{s}^n} p_k(\tilde{s}^n|s^n, a^n) \tilde{V}^*(\tilde{s}^n) \Big\}, \quad (6)$$

where $\mathcal{A}$ is set of actions and $\gamma \in [0, 1)$ is the discount factor. The optimal policy is then evaluated as

$$\Pi^*_{PDS}(s^n) = \arg \max_{a^n \in \mathcal{A}} \Big\{ r_k(s^n, a^n) + \gamma \sum_{\tilde{s}^n} p_k(\tilde{s}^n|s^n, a^n) \tilde{V}^*(\tilde{s}^n) \Big\}. \quad (7)$$

The value of the PDS state $\tilde{s}^n$ in terms of the next state s^{n+1} is given by (5) and the value of the state s^n in terms of the PDS state $\tilde{s}^n$ is given by (6). Since the dynamics are known in the state transition from s^n to $\tilde{s}^n$ and the actions taken do not affect the transition from $\tilde{s}^n$ to s^{n+1}, we can use the greedy algorithm (indicated by the max function) to find the optimal actions that can be taken at a given time instant n, as seen in (6). In other words, since the unknown dynamics are independent of the actions taken, no exploration-exploitation routine is necessary to find the optimal policies.

The PDS-based online learning algorithm is summarized below [18]

$$V^n(s^{n+1}) = \max_{a \in \mathcal{A}} \{ r_k(s^{n+1}, a) + \sum_{\tilde{s}} p_k(\tilde{s}|s^{n+1}, a) \tilde{V}^n(\tilde{s}) \}$$

$$\tilde{V}^{n+1}(\tilde{s}^n) = (1 - \alpha^n) \tilde{V}^n(\tilde{s}^n) + \alpha^n \gamma V^n(s^{n+1}) \quad (8)$$

***Theorem** 1:* The post decision state-based learning algorithm converges to the optimal post decision state value function $\tilde{V}^*(\tilde{s})$ when the sequence of learning rate α^n satisfies $\sum_{n=0}^{\infty} \alpha^n = \infty$ and $\sum_{n=0}^{\infty} (\alpha^n)^2 = \infty$.

Proof: Due to space limitations, we only sketch the proof here. For each PDS $\tilde{s}$, define $F_{\tilde{s}}(\tilde{V}) = \max_{a \in A}(r(s, a) + \gamma \tilde{V}(\tilde{s}))$, where s and $\tilde{s}$ satisfy the relationship defined earlier. $F : \mathbb{R}^{|\mathcal{S}|} \rightarrow \mathbb{R}^{|\mathcal{S}|}$ is a mapping such that $F(\tilde{V}) = [F_{\tilde{s}}(\tilde{V})]_{\tilde{s}}$ (i.e., for all PDS). From [17]-[19], the convergence of the proposed algorithm is equivalent to the associated O.D.E given by $\dot{\tilde{V}} = F(\tilde{V}) - \tilde{V}$. Since F is a maximum norm γ-contraction [17], the stability of the unique equilibrium point of the O.D.E and thereby the proposed algorithm (which corresponds to $V^*(\tilde{s})$) are guaranteed [18]. This further enables to learn the optimal value function $V^*(s)$ and the optimal policy Π^*_{PDS}.

IV. Numerical Results

A. Performance improvement in the single-user setting

We first discuss the learning results in single-user settings. Though we do not consider the various physical layer parameters that are involved in the 802.11 standard, notice that the impact of channel conditions and thereby the transmit power, modulation and coding constraints can be easily included in this framework by following the formulation in [19]. Unless otherwise specified, we have $B = 25$, $CW_{\min} = 2$, $CW_{\max} = 16$, the arrival distribution is taken to be Poisson with mean arrival rate 1 packet/second, and the discount factor $\gamma = 0.98$. The reward on throughput $T = 100$, the energy cost $E = 5$, and the collision penalty $C = 5$ (the behavior of the algorithms is independent of the absolute values of these parameters).

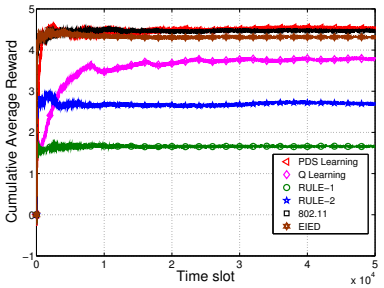

Fig. 2. Cumulative average reward obtained using various algorithms in the presence of low network load ($\rho = 0.1$), $T = 100, E = 5, C = 5$, and $PLR = 0.1$.

Figs. 2 and 3 show the performance of the various contention access algorithms in terms of the cumulative average reward when $PLR = 0.1$ and the collision probability $\rho = 0.1$ and $\rho = 0.9$ respectively. These cases capture the scenarios where the network load is low and high respectively. Here we show the performance of the PDS learning algorithm and the traditional Q-learning algorithm along with three rule-based (deterministic policies) approaches, which are a) randomly choosing a back-off in the RTS COLLISION and CTS RECEIVED states (RULE-1), b) choose the minimum back-off window size when the packet is delivered successfully and the maximum window size when there is a packet failure or a RTS collision (RULE-2) and c) choose the back-off window size in between $CW_{\min}$ and $CW_{\max}$ as per the 802.11 protocol and d) exponential increase exponential decrease (EIED) algorithm [7] which was proposed as an alternative to the 802.11 DCF mechanism, especially when the network load is high. In EIED, the back-off window exponentially increases by 2 when there is a collision or a packet failure and exponentially decreases by 2 when there is a packet success.

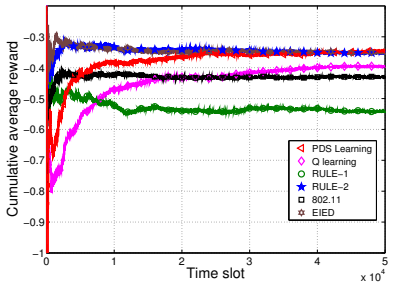

Fig. 3. Cumulative average reward obtained using various algorithms in the presence of high network load ($\rho = 0.9$), $T = 100, E = 5, C = 5$, and $PLR = 0.1$.

When the network load is low *i.e.*, $\rho = 0.1$, it is known that the 802.11 protocol performs well [2], [7] and hence high rewards are achieved as seen in Fig. 2. Notice that the PDS learning algorithm performs as well as the 802.11 protocol and the EIED algorithm but the Q-learning algorithm is worse in comparison to these algorithms. While the PDS algorithm achieves higher rewards by exploiting the knowledge of the known dynamics of the environment and thereby avoids action exploration, the Q-learning algorithm performs worse because of its exploration-exploitation procedure. Also, as expected, these algorithms perform much better than the other deterministic rule-based approaches.

When the network load is high *i.e.*, $\rho = 0.9$, the sub-optimality of the 802.11 protocol is well-understood [2], [7] and is in agreement with the results seen in Fig. 3. Interestingly it is seen that while the PDS learning algorithm performs better than the Q-learning algorithm and RULE-1, it approaches the performance of the RULE-2 algorithm. As discussed in [7], the EIED algorithm performs better than the 802.11 protocol when the network load is high and is close to the performance achieved by the PDS algorithm. The PDS, EIED and RULE-2 algorithms consistently choose higher back-off window values which explains the reason for their similar performance.

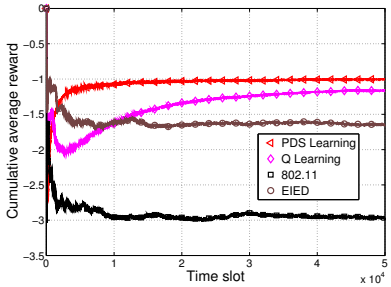

Fig. 4. Cumulative average reward obtained using various algorithms in the presence of moderate network load ($\rho = 0.5$), $T = 100, E = 10, C = 10$, and $PLR = 0.1$.

Fig. 4 shows the performance of the various algorithms when $\rho = 0.5$ *i.e.*, the case of a medium network load. While the EIED algorithm outperforms the 802.11 protocol [7], the PDS learning and the Q-learning algorithms perform significantly better than both of them. Thus the true benefit of the learning algorithms over most heuristic algorithms is seen in the presence of medium network loads. In addition to the higher rewards obtained by the PDS learning algorithm, it is seen that it converges much faster than the Q-learning algorithm. This improved performance is due to the splitting of the environment dynamics into the known and unknown components and avoiding the action exploration phase as discussed earlier.

Fig. 5 shows the back-off window values chosen by the PDS and Q-learning algorithms. While the PDS learning algorithm converges fast to the optimal policy (here, policy refers to choosing the optimal back-off window size), the Q-learning algorithm does not converge even after a long time as the exploration phase depends on the number of times a state and action pair has been visited. This is because the optimality of the Q-learning algorithm is guaranteed only when all the possible state-action pairs are visited infinitely many times. The mean back-off window sizes chosen by the PDS learning, Q-learning and the 802.11 protocols are shown in Table II as a function of ρ. As expected, the back-off window size is a monotonically increasing function of the collision probability ρ (as network load increases, the user has to wait for longer time intervals to avoid any potential collisions with the other users).

TABLE II. MEAN BACK-OFF WINDOW SIZE AS A FUNCTION OF COLLISION PROBABILITY ρ.

ρ	Mean Window Size (PDS)	Mean Window Size (Q-Learning)	Mean Window Size (802.11)
0.1	2.01	2.45	2.14
0.5	7.45	10.71	3.84
0.9	13.32	13.94	12.27

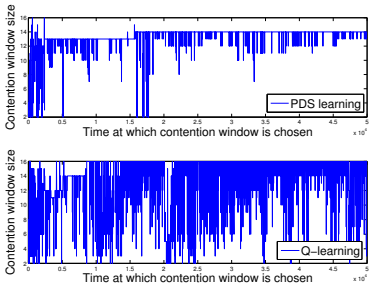

Fig. 5. Back-off window sizes chosen at the RTS COLLISION and CTS RECEIVED protocol states, $T = 100, E = 5, C = 5, \rho = 0.9, PLR = 0.1$.

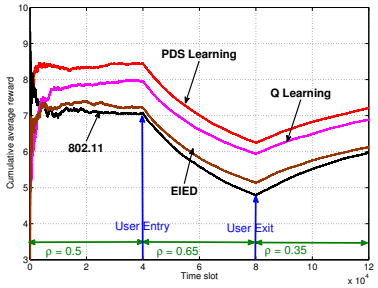

Fig. 6. Cumulative average reward obtained using various algorithms in the presence of varying network load, $T = 100, E = 0, C = 10, PLR = 0.1$.

Fig. 6 shows the performance of the various algorithms in the presence of a varying network load (users enter and leave the network randomly) which is modeled by varying ρ. While the cumulative average reward obtained in all these cases behaves similarly (the rewards obtained by the various algorithms is in accordance with the changes in ρ because collisions increase when ρ increases and leads to higher penalty when sub-optimal back-off window sizes are chosen), it is seen that the PDS algorithm has an advantage because it exploits the structure of the network at every time instant, which cannot be done in the other learning or heuristic algorithms. Overall, the advantage of the PDS algorithm is clearly seen from the Figs. 2-6 and Table II.

B. Multi-user setting

Consider an 802.11 network with multiple users each of which is selfish and intends to maximize its own rewards. In this case, the collision probability parameter ρ used in the single user setting is no longer relevant as the collisions now occur based on the decisions taken by each user. Further, each user is intelligent and employs learning algorithms in an attempt to learn the environment and optimizes its actions in order to maximize its own rewards. As opposed to many centralized algorithms, there is no message exchange or cooperation between these users and is hence a truly decentralized setting. Solving a multi-user MDP problem is difficult in such decentralized settings, especially in cases where the MDP of each user is coupled with the others (because one user's actions affect the state transitions of the other users). We thus study the performance of the single user learning algorithms developed earlier when extended to these multi-user settings.

Fig. 7 shows the average reward achieved using different contention algorithms in a wireless network with 10 users. Again in this case, the PDS algorithm has a superior performance in comparison with other algorithms. While the mean (across all users) cumulative average (over the time

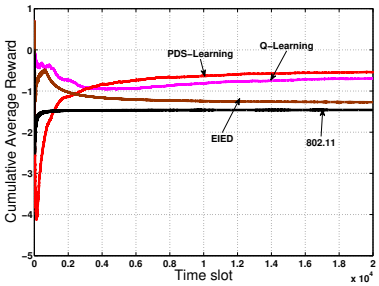

Fig. 7. Cumulative mean reward obtained using various MAC algorithms in a wireless network with 10 users, $T = 100$, $E = 5$, $C = 30$, and $PLR = 0.1$.

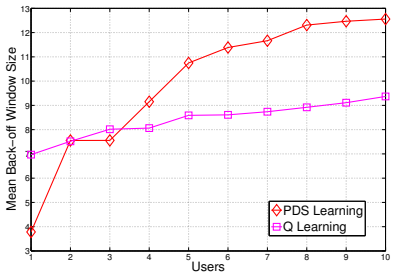

Fig. 8. Sorted mean back-off window sizes of the 10 users in a wireless network, each using the PDS or Q-learning algorithms.

horizon) reward achieved by the PDS algorithm is higher, it was observed that it is not fair in terms of the users' wireless channel accessibility, i.e., some users access the channel more compared to others as shown in Fig. 8. The superior performance of PDS algorithm over Q-learning is also evident from the mean back-off window sizes shown in Fig. 8. Since the mean back-off window sizes of the users are close to each other in the case Q-learning when compared to PDS, more collisions can occur which also explains the lower average rewards seen in Fig. 7.

The unfairness in the network utilization when using these learning algorithms depends on the back-off window sizes chosen by the users. For example, consider a network with 2 users, each of which is using the above algorithms to learn the back-off window sizes. Depending on which user picks a smaller back-off window first (since the users try various actions), the other user will automatically be forced to choose a higher back-off window in order to avoid collisions and the penalties incurred. This happens because the value functions of the PDS and Q-Learning algorithms consider the future effects of their action choices as seen in (5), (6) and (7). Thus, they are pessimistic in their action choices so as to avoid any impending collisions. This in turn results in an unfair network allocation to the users (while the overall reward can still be high as seen in Fig. 7). These fairness issues are unavoidable, especially in the case of decentralized multi-user systems where there is no means of communicating with other users or there is no centralized authority to allocate the resources. Investigating decentralized learning algorithms that are fair to all the users but do not need any message exchange is part of our ongoing work.

V. Conclusion

In this paper, we showed via RL algorithms that the contention-based MAC access protocols can be improved in terms of the network efficiency by appropriately choosing the back-off window. Firstly, for the single-user learning setting, by using the post-decision state-based learning algorithm, we exploited the the structure of the problem by separating the known and the unknown components of the environment. This improved the performance when compared to Q-learning based algorithms and other heuristic policies widely used in 802.11-based networks. The results suggest that a smaller back-off window size is to be chosen when the network load is smaller and vice versa. Fairness issues arise when these single user algorithms are extended to multi-user settings where there is no communication between different users. Investigating fair multi-user learning algorithms in the context of MAC-layer access protocols is an interesting avenue to pursue.

References

[1] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 3, pp. 535-547, Mar. 2000.

[2] J. G. Andrews *et al.,* "Fundamentals of WiMAX: Understanding Broadband Wireless Networking". Upper Saddle River, NJ, Prentice-Hall, 2007.

[3] A. Ksentini *et al.,* "Determinist contention window algorithm for IEEE 802.11," in *Proc. PIMRC*, vol. 4, Sept. 2005, pp. 2712-2716.

[4] A. Nafaa *et al.,* "SCW: Sliding Contention Window for Efficient Service Differentiation in IEEE 802.11 Networks", in *Proc. WCNC*, New Orleans, LA, Mar. 2005, pp. 1626-1631.

[5] Q. Pang *et al.,* "A TCP-like adaptive contention window scheme for WLAN," in *Proc. ICC*, vol. 6, Jun. 2004, pp. 3723-3727.

[6] Y. Kwon *et al.,* "A novel MAC protocol with fast collision resolution for wireless LANs," in *Proc. INFOCOM*, vol. 2, Apr. 2003, pp. 853-862.

[7] N. O. Song *et al.,* "Enhancement of IEEE 802.11 distributed coordination function with exponential increase exponential decrease back-off algorithm," in *Proc. IEEE VTC*, New Orleans, Apr. 2003, pp. 2775-2778.

[8] N. Choi *et al.,* "P-DCF: enhanced back-off scheme for the IEEE 802.11 DCF," in *Proc. VTC*, Stockholm, vol. 3, Jun. 2005, pp. 2067-2070.

[9] M. Ghazvini *et al.,* "Game Theory Applications in CSMA Methods," *IEEE Commun. Surv. and Tut.*, vol. 15, no. 3, pp. 1062-1087, Jul. 2013.

[10] Y. Xiao *et al.,* "Game theory models for IEEE 802.11 DCF in wireless ad hoc networks, *IEEE Commun. Mag.*, vol. 43, no. 3, pp. S22-S26, Mar. 2005.

[11] P. Krishna, *et al.,* "Virtual backoff algorithm: an enhancement to 802.11 medium-access control to improve the performance of wireless networks," *IEEE Trans. Veh. Technol.,* vol. 59, no. 3, pp. '068-1075, Mar. 2010.

[12] T. Abdelkader and K. Naik, "A localized adaptive strategy to calculate the backoff interval in contention-based vehicular networks," *IEEE Access.,* vol. 2, pp. 215-226, Mar. 2014.

[13] M. Levorato *et al.,* "Cognitive Interference Management in Retransmission-Based Wireless Networks,", *IEEE Trans. Inf. Theory*, vol. 58, no. 5, pp. 3023-3046, May. 2012.

[14] J. Barcelo *et al.,* "Learning-BEB: avoiding collisions in WLAN," in *Proc. Eunice Summer School,* Brest, France, Sept. 2008, pp. 1-8.

[15] M. Fang *et al.,* "Decentralised learning MACs for collision-free access in WLANS," *Wireless Netw.,* vol. 19, no. 1, pp. 83-98, May 2012.

[16] W. Zame *et al.,* "Winning the Lottery: Learning Perfect Coordination with Minimal Feedback," in *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 5, pp. 846-857, Oct. 2013.

[17] N. Salodkar *et al.,* "An On-Line Learning Algorithm for Energy Efficient Delay Constrained Scheduling over a Fading Channel," *IEEE J. Sel. Commun.*, vol. 26, no. 4, pp. 732-742, Apr. 2008.

[18] F. Fu and M. van der Schaar, "Structure-Aware Stochastic Control for Transmission Scheduling," *IEEE Trans. Veh. Tech.,* vol. 61, no. 9, pp. 3931-3945, Nov. 2012.

[19] N. Mastronarde and M. van der Schaar, "Joint Physical-Layer and System-Level Power Management for Delay-Sensitive Wireless Communications," *IEEE Trans. Mobile Comput.*, vol. 12, no. 4, pp. 694-709, Apr. 2013.