perMAC: Perturbation-based MAC for Dense Wireless Networks with Periodic Traffic

Jing Deng
Dept. of CS
UNC Greensboro
Greensboro, NC 27412, U.S.A.
jing.deng@uncg.edu

Po-Ning Chen
Dept. of ECE
National Yang Ming Chiao Tung Univ.
R. O. C.
poningchen@nycu.edu.tw

Yunghsiang S. Han
Shenzhen Inst. for Advanced Study
Univ. of Electronic Sci. and Tech. of China
Shenzhen, P. R. China
yunghsiangh@gmail.com

***Abstract*— The abundance of wireless devices necessitates efficient contention resolution among competing users. With different approaches and channel reservation techniques in use, the ultimate resolution approach still remains similar to pure ALOHA or slotted ALOHA. It has been proven that no approach could achieve a better throughput than pure ALOHA or slotted ALOHA if users are randomly competing with each other. In this work, we investigate an efficient mechanism to improve contention resolution among competing users in dense wireless networks with periodic traffic. Our approach is to use transmission perturbation among competing users in order to avoid repeated collisions down the road. Analysis and performance evaluations show that our new design, Perturbation-based MAC (perMAC), achieves a high throughput and is rather stable.**

I. Introduction

We live in a networked world, even for those of us located in rather remote areas. With the proliferation of wireless devices competing for the limited wireless communication resources, collisions tend to happen frequently, which lower system throughput. Therefore, a carefully-designed technique is needed to allow these users to access the shared channel.

The large number of wireless devices competing for the same resource results in a waste of its time with all competing users. With ever-more popular IoT devices, more and more users are joining the competition. Even with the implementation of rather advanced techniques, these users' competition would still be capped by theoretical bounds of pure ALOHA technique, or in the case of slotted operations, slotted ALOHA. Essentially, when pure ALOHA is used, the best throughput that can be achieved is $1/2e = 0.184$ and for slotted ALOHA, it is $1/e = 0.368$.

The question remains open whether it is possible to somehow improve the throughput of such systems. In this work, we approach this problem from the angle of many realistic applications, i.e., most devices are on periodic traffic [1] even though these traffic flows are rather unpredictable. Many of the data demanding 6G communications such as Virtual Reality (VR), Augmented Reality (AR), massive connectivity such as IoT networks connecting billions of machine-type devices, ubiquitous wireless devices, could very well fall in this category. That is to say, these users need to send data packets periodically and they need to send these within a certain delay bound (expired packets should be discarded because new ones might have been generated already.) With such a large number contending nodes, traditional MAC schemes might not work, leading to the need for Next Generation Multiple Access (NGMA) design.

In this work, we design a new approach, termed Perturbation-based MAC (perMAC), to help improve the throughput. Our approach is based on the following observations: when periodic traffic is concerned, they only need a specific transmission time slot with as low competition as possible. With the help from a novel approach of perturbing the transmission schedules of these nodes, we should be able to "spread" them quite well in each of the cycles. Note that this simple approach can be used on top of most existing protocols without major changes.

II. Related Work

Periodic traffic in wireless networks is not unheard of. In fact, many published works have investigated wireless networks [1]–[6]. Many of these achieved throughput better than ALOHA or slotted ALOHA, though with various assumptions.

Cao et al. presented an analytical model and some extensions that accounted for heterogeneous traffic and hidden nodes in [1]. Yuan et al. [2] focused on intelligent Wireless Body Area Networks (WBAN) and its real-time and reliable health data transmissions. A health critical index was designed to prioritize different traffic flows' transmissions. In [3], Mennes et al. took advantage of neural networks to perform online learning in the process of identifying free slots in Multiple Frequency TDMA (MF-TDMA) networks. By doing this, collisions could be reduced 15 times. Ahmed and Hussain proposed a scheduling technique in IEEE 802.11ah network. Their approach predicts the service interval of monitoring applications and schedules subsequent frame transmissions before they arrive, reducing delay and unnecessary early wake-up (and thus energy consumption). In [5], Lusvarghi and Merani studied cellular vehicle-to-everything (C-V2X) communications mode 4, which could see the coexistence of periodic traffic and aperiodic traffic. Yousefi et al. studied time critical wireless sensor networks under periodic traffic with reneging packets through a Markovian chain model [6].

MAC designs for dense wireless networks have also appeared in technical literature [7], [8]. Narasimha et al. used

Mean Field Game (MFG) to analyze wireless networks in the large population regime. Mean Field Nash equilibrium and price of anarchy were investigated. Gao et al. [8] designed a MAC scheme for a massive number of devices with sporadic data traffic.

Frameless ALOHA design [9] and several subsequent applied paradigm of rateless codes in the design of MAC schemes, allowing the frame length to be adjustable while users adjust their transmission probabilities. The approach achieved a throughput significantly higher than slotted ALOHA for non-periodic setting.

The approach we are presenting in this work is also related to several machine learning approaches [10]–[13]. Ali et al. [11] used deep reinforcement learning to optimize throughput for observation-based MAC scheme in Wireless LAN (WLAN). Zhang et al. [12] used coordinated descent federated learning on the selection of simultaneous transmission over k sub-carriers in order to achieve higher reception success at the common receiver in the many-to-one communication setting. Cordeschi et al. [13] proposed a Delay-Collision CSMA (DC-CSMA) scheme in order to reduce user access latency and to preserve high success ratio at the same time.

Compared to these related works, our approach is to focus on the shared channel with a large number of active users with periodic traffic. We try to spread their periodic transmissions over the transmission period in order to improve overall system performance. Using this approach, we are able to achieve throughput that is more than double of slotted ALOHA.

III. System Model and Design

A. System Model

In the system that we study, we assume there are N users with traffic to send to a centralized receiver, unless specified otherwise. These users use slotted communication, i.e., transmissions are all synchronized with the start time of each slot. We call each slot a unit time. The transmission delay is assumed to be negligible, although further investigations of non-negligible transmission delays would be interesting and can serve as direction for future work. Each user is able to observe its own transmissions' success, either through off-channel broadcast from the centralized receiver or by a short acknowledgment message. Furthermore, users are able to estimate the number of users N in the network [14].

User traffic is not pure random. Instead, all users have the same delay/arriving period D but they could start at different times/offsets. We investigate dynamic traffic in the performance evaluation section and leave different traffic patterns to our future work.

B. Discussions and Design

In our pursuit of possible throughput higher than the well-known bound of slotted ALOHA's $1/e$, we notice that the delay-constrained traffic pattern among all competing video streams and the synchronized frame assumption has been made for the convenience of analysis. We also hope to understand the performance of ALOHA-type protocols for such non-frame-synchronized traffic pattern.

In this subsection, we consider a randomly-offset version of the frame-synchronized traffic pattern: all stations have the same delay/arriving period D but could start at different times/offsets, denoted as

$$\mathbf{\Gamma} = (\Gamma_1, \Gamma_2, \cdots, \Gamma_N), \tag{1}$$

where any Γ_i is a uniform random variable in the range of $\{1, 2, \cdots, D\}$ and all Γ_i's are i.i.d. The first packet of flow k arrives at the beginning of slot Γ_k. Clearly, our frame-synchronized traffic pattern is a special case of non-frame-synchronized traffic pattern, with

$$\mathbf{\Gamma} = (\Gamma_1, \Gamma_2, \cdots, \Gamma_N) = (1, 1, \cdots, 1). \tag{2}$$

First, it is straightforward to show that earliest-deadline-first (EDF) [15], maximizes the system throughput for such delay-constrained traffic, but EDF requires a central controller. Therefore, we aim to design ALOHA-type decentralized protocol. To do this, we try to approximate the EDF scheduling policy in a decentralized manner. In decentralized techniques, nodes are unlikely to know what other nodes' packet deadlines.

A careful observation on the competing streams is thus needed: in realistic applications, shared streams start randomly. For the benefit of easy presentation, we assume that the data packets in different streams have the same delay constraints, i.e., they expire in the same amount of time, modeled as D here. However, due to the randomness of each stream's start times, clusters of start times are bound to take place. In fact, the chance of an evenly distributed start time can be computed as [16]

$$p_0(D, N) = \frac{\frac{D!}{(D-N)!}}{D^N}\ , \tag{3}$$

where the numerator in (3) represents the number of different arrangements of all N nodes' start times with no collisions and the denominator in (3) is the total number of different arrangements of all N nodes' possible start times. As an example, when D=100 and N=40, $p_0(D, N) = 0.01\%$.

However, it is more likely that these start times are rather spread out relatively evenly. Using the same (D, N) set values of (100, 40), we show one set of randomly generated start times on Table I. Those start times overlapping with other users are underlined and shown in red. Altogether, 13 (marked in red) out of the N=40 users have such overlaps and they reside on 6 spots. It means, the rest of 27 users should be able to send at their deadline spot without any collisions. The issue is how to resolve the ones with collisions. A distributive strategy is "perturbation," [17], [18] discussed below.

Denote ℓ_i as the number of slots when user i's packet (if any) expires. Clearly, $\ell_i \in \{1, 2, \cdots, D\}$. We would simply call ℓ_i the *lead time* of station i [19]. The relationship between ℓ_i and Γ_i at time slot t can be expressed as

$$\ell_i = \mod(\Gamma_i - t, D)\ . \tag{4}$$

EDF schedules the station i^* with the minimal expiration

TABLE I
EXAMPLE OF RANDOM START TIMES FOR (D,N)=(100,40)

9	10	15	16	16	17	19	21	22	25	25	25	26	26	35	36	39	41	44	47
48	55	59	59	60	62	62	63	63	64	66	72	77	80	84	87	88	89	90	95

time, i.e.,

$$i^* = \arg\min_j l_j \tag{5}$$

and breaks ties arbitrarily. Instead, we try to design a distributed ALOHA scheme where each station only knows D, N, its packet's lead time, among other observable information.

One design is to favor those data packets with short lead times by setting their transmission probabilities arbitrarily high:

$$p(\ell_i) = \begin{cases} \min\{\frac{\alpha}{m}, 1\}, & \text{if } \ell_i \le m; \\ \min\{\frac{D/N-\alpha}{D-m}, 1\}, & \text{otherwise,} \end{cases} \tag{6}$$

where $p(\ell_i)$ is the retransmission probability when the leading time is $\ell_i = 1, 2, \cdots, D$ for user i, $m \in \{1, 2, \cdots N\}$ and $\alpha > 0$, although m is expected to be an integer close to 1 and α should not be larger than 1 (this was explained in [14] and [20] in details).

Unfortunately, transmission policy (6) usually fails to deliver high throughput because of the higher chances of having deadline (or lead time) collisions, i.e., nodes with packets that are set to expire in the same time slot. Such packets have been set by the transmission policy to have much higher transmission probability and they will collide with each other repeatedly.

In order to reduce the chance of lead time collisions, we design a technique called Perturbation-based MAC (perMAC) to address the problem. The design came from an observation is that, after transmission policy (6) is employed for some time, each node is able to find out how much throughput has gone through from itself over the competing wireless channel. This can be used to guess whether repeated collisions have been taking place or not.

In perMAC, in every $T \gg D$ time slots, each node compares its throughput S_i (the total number of successfully transmitted packets divided by time) with θ/N, where $0 < \theta < 1$ is the triggering threshold. Those nodes suffering with low throughput share will adjust their lead times within the range of $\pm R$ time slots as a measure to avoid repeated collisions. The algorithm is described in Algorithm 1, which is executed every T unit times by the nodes.

The reason for this design is that users colliding frequently should be switched or perturbed to other spots for better results. By allowing these users to make small shifts gradually, i.e., perturbations, the perMAC scheme eventually spread these competing users into the entire period of D, resulting in higher throughput. Note that Algorithm 1 requires user's knowledge of N, which can be estimated [14]. We further investigate the impact of inaccurate N estimate on perMAC's performance in Section V.

Algorithm 1 perMAC Lead Time Adjustment Algorithm

```
Require: $S_i$, $N$, $\ell_i$, $i = 1, 2, \cdots, N$
Ensure: New lead time $\ell'_i$, $i = 1, 2, \cdots, N$
1: for all $i \in \{1, 2, \cdots, N\}$ do
2:     if $S_i < \theta/N$ then
3:         $\ell'_i = [\ell_i + \lfloor 2R * rand() \rfloor - R] \mod D$
4:     else
5:         $\ell'_i = \ell_i$
6:     end if
7: end for
```

IV. ANALYSIS

We use Poisson arrival to estimate the throughput of the perMAC scheme in this section. For simplicity, here we assume that users' choices of different time slot follow Poisson distribution. At the very beginning, there are N users that will randomly pick one of the D time slots. Only those users picking a time slot by themselves will be successful. Other colliding users will perform permutation and hope to be successful later.

The expected number of successful picks among all N users is

$$\eta_{i=1} = D \cdot \text{Pr(one arrival)} = D\frac{N}{D}e^{-N/D} = Ne^{-N/D} \tag{7}$$

which calculates the chance of success in each of the D time slots in the first round (before any perturbation, $i = 1$). The Poisson arrival rate is estimated as N/D.

Any subsequent round of perturbation will require those users that have perturbed but still haven't identified a success time slot to continue their perturbations. The expected number of success among these users is

$$\eta_k = (D - \sum_{i=1}^{k-1} \eta_i) \cdot \text{Pr(one arrival)} = (N - \sum_{i=1}^{k-1} \eta_i) e^{\frac{(N-\sum_{i=1}^{k-1}\eta_i)}{(D-\sum_{i=1}^{k-1}\eta_i)}} \tag{8}$$

in which we have removed those successful users in all previous rounds, a total number, $\sum_{i=1}^{k-1} \eta_i$, of them. These are users with a single time slot to themselves. Note that, even though occasionally some other new users would perturb and join these time slots, they would move away noticing the new collisions while the previously successful users would stay.

For K rounds, we count throughput as the sum of all successful time slot selections: $S = \sum_{k=1}^{K} \eta_k$. We compare our analysis with simulation results in Section V.

V. PERFORMANCE EVALUATION

Extensive simulations have been performed to analyze the throughput of perMAC. In our study, we focus on throughput,

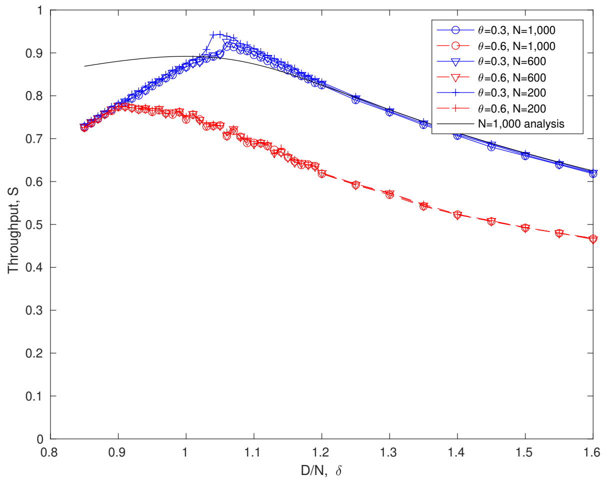

Fig. 1. Throughput of perMAC with different δ, θ, and N.

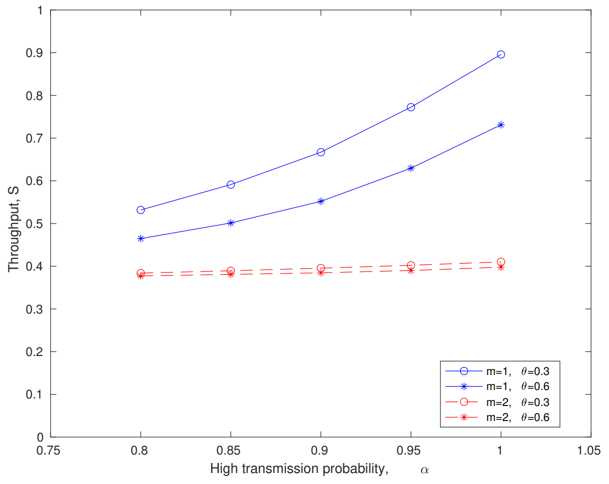

Fig. 2. Throughput for perMAC with different α, high transmission probability ($N = 1,000$.)

number of colliding users, energy consumption (in terms of attempted transmissions), packet drop rate, and standard deviation of throughput among all active users (for fairness investigation). All simulations were implemented in MATLAB and run for 20 rounds with average calculated from these results. Simulation time was 1,000,000 units, observation period $T = 10,000$, $R = \lfloor 0.1 \times D \rfloor$, and $\alpha = m = 1$, unless specified otherwise.

First, we investigated the impact of $\delta = D/N$ and θ on the throughput performance of the perMAC scheme. In Figure 1, we used two different θ values and a wide range of δ. We chose a few N values, 200, 600, and 1,000 in our evaluation. Obviously other N values can be used and the conclusions should be similarly. It can be seen that θ should be relatively small for perMAC to function well. This is because of the adverse impact of a large θ and the over-aggressive perturbation of user's lead time (note that users with a recent throughput smaller than the θ/N threshold would perform the perturbation of their lead times. With respect to δ, perMAC performance improves generally with the increase of δ until reaching its peak and then it will level off. When δ is too small, e.g., $\delta < 1$ meaning $D < N$, there is not enough spots for all the competing users. However, when δ is increased beyond 1, more spots are eventually wasted as the number of active user is now smaller than the number of available spots.

Analysis results are also presented in Figure 1 as the solid dark line (with $K = 20$). While our simple analysis failed to match the throughput trend for smaller δ, it did highlight the throughput close to 0.9 without considering θ and match simulation throughput for larger δ well.

In order to illustrate this point further, we have shown throughput for different m values (1 and 2) with a range of α as the high transmission probability and different θ in Figure 2. It can be observed that, when $m = 2$, throughput change is almost negligible when we change θ from 0.3 to 0.6. However, whenever $m = 1$, a significant throughput drop could be seen when θ is changed from 0.3 to 0.6.

TABLE II
DROPPED RATIO OF THE PERMAC SCHEME

	δ=0.85	δ=0.95	δ=1.05	δ=1.15	δ=1.35
N=1,000	0.4491	0.2308	0.0535	0.0103	0.0082
N=400	0.4480	0.2271	0.0415	0.0064	0.0047
N=200	0.4432	0.2237	0.0055	0	0.0019

Conversely related to the throughput is the ratio of dropped packets (among all generated packets), $P_d = \frac{1}{\delta} - S$. Results are shown on Table II. We can see that the dropped ratio decreases as δ increases towards 1 and then it will increase slightly, suggesting that operating δ at slightly large values would not hurt dropped ratio. When δ is about 1.05 or 1.15, less than 5% packets were dropped in the perMAC scheme.

Users should have their fair share of channel usage. We showed the standard deviation of throughput among different nodes on Table III. These were computed based on each user's overall throughput over the entire simulation time. The observed values are very small and they tend to decrease with N (more evenly distributed among large number of nodes) and δ (more spread out and easier to transmit packets successfully).

We were also interested in perMAC's performance when the number of nodes with actual traffic changed over time. These results are shown in Figure 3. During the first third of the 300,000 total simulation time, i.e., first 100,000 time slots, half of the N nodes in the network generated packets for transmissions. During the second third, i.e., from 100,000

TABLE III
STANDARD DEVIATION OF THROUGHPUT AMONG ALL USERS ($\times 10^{-3}$)

	δ=0.85	δ=0.95	δ=1.05	δ=1.15	δ=1.35
N=1,000	0.0812	0.0667	0.0336	0.0103	0.0082
N=400	0.1058	0.0875	0.0386	0.0105	0.0079
N=200	0.1935	0.1581	0.0224	0.0111	0.0093

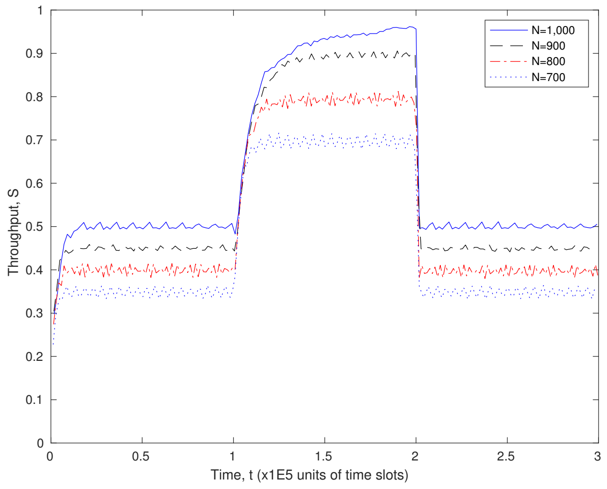

Fig. 3. Performance of the perMAC scheme with dynamic number of nodes with traffic. $D = 1,000$ in these simulations.

to 200,000, all N nodes would generate packets for transmissions. And during the last third time, i.e., after 200,000 until the end of the simulation, only half of the N nodes would generate packets for transmissions again. We used this simple dynamics in traffic load to evaluate the performance of the perMAC scheme. Note that this is the only graph that shows results with some nodes in the network having no packets to send. Also note that D was set to 1,000 in all of these simulations (this is different to other simulations) and the same N, i.e., 1,000, 900, 800, and 700, were used throughout the simulation even though some nodes would not have any traffic over a period of time.

Our first observation is that, when half of the N nodes had traffic, all generated packets were transmitted successfully, with overall throughput of 0.5, 0.45, 0.4, and 0.35 for $N = 1,000$, 900, 800, and 700 (note that $D = 1,000$). More interesting is the increase of perMAC's throughput during the middle third of the simulation time. For $N = 800$ and 700, the throughput would increase to 0.8 and 0.7 quickly and stays there until traffic load drops. For larger N (1,000 and 900, i.e., closer to D), this increase is rather slow. In fact, the throughput could never reach 1 when $N = 1,000$ (it reaches about 0.94). When $N = 900$, throughput reaches 0.9 safely. These results show that the perMAC scheme reacts to the different traffic load and provides support for user traffic as needed.

VI. Concluding Remarks

In this work, we have focused on the MAC design issues for densely populated wireless networks where data traffic are unpredictable although periodic by each active user. Taking advantage of the periodic traffic, we have designed a new MAC scheme, termed Perturbation MAC (perMAC), to improve throughput beyond slotted ALOHA, on which our scheme operates. Our extensive simulation results show how and why perMAC is able to offer its superior performance compared to other techniques.

References

[1] X. Cao, J. Chen, Y. Cheng, X. S. Shen, and Y. Sun, "An analytical mac model for ieee 802.15.4 enabled wireless networks with periodic traffic," *IEEE Transactions on Wireless Communications*, vol. 14, no. 10, pp. 5261–5273, 2015.

[2] X. Yuan, C. Li, Q. Ye, K. Zhang, N. Cheng, N. Zhang, and X. Shen, "Performance analysis of ieee 802.15.6-based coexisting mobile wbans with prioritized traffic and dynamic interference," *IEEE Transactions on Wireless Communications*, vol. 17, no. 8, pp. 5637–5652, 2018.

[3] R. Mennes, M. Camelo, M. Claeys, and S. Latré, "A neural-network-based mf-tdma mac scheduler for collaborative wireless networks," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, 2018, pp. 1–6.

[4] N. Ahmed and M. I. Hussain, "Periodic traffic scheduling for ieee 802.11ah networks," *IEEE Communications Letters*, vol. 24, no. 7, pp. 1510–1513, 2020.

[5] L. Lusvarghi and M. L. Merani, "On the coexistence of aperiodic and periodic traffic in cellular vehicle-to-everything," *IEEE Access*, vol. 8, pp. 207 076–207 088, 2020.

[6] H. H. N. Yousefi, Y. Kavian, and A. Mahmoudi, "A markov chain model for ieee 802.15.4 in time critical wireless sensor networks under periodic traffic with reneging packets," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, pp. 2253–2268, 2022.

[7] D. Narasimha, S. Shakkottai, and L. Ying, "A mean field game analysis of distributed mac in ultra-dense multichannel wireless networks," *IEEE/ACM Transactions on Networking*, vol. 28, no. 5, pp. 1939–1952, 2020.

[8] J. Gao, W. Zhuang, M. Li, X. Shen, and X. Li, "Mac for machine-type communications in industrial iot—part i: Protocol design and analysis," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9945–9957, 2021.

[9] C. Stefanovic, P. Popovski, and D. Vukobratovic, "Frameless aloha protocol for wireless networks," *IEEE Communications Letters*, vol. 16, no. 12, pp. 2087–2090, 2012.

[10] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of machine learning in wireless networks: Key techniques and open issues," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3072–3108, 2019.

[11] R. Ali, N. Shahin, Y. B. Zikria, B.-S. Kim, and S. W. Kim, "Deep reinforcement learning paradigm for performance optimization of channel observation–based mac protocols in dense wlans," *IEEE Access*, vol. 7, pp. 3500–3511, 2019.

[12] J. Zhang, N. Li, and M. Dedeoglu, "Federated learning over wireless networks: A band-limited coordinated descent approach," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, 2021, pp. 1–10.

[13] N. Cordeschi, F. De Rango, and M. Tropea, "Exploiting an optimal delay-collision tradeoff in csma-based high-dense wireless systems," *IEEE/ACM Transactions on Networking*, vol. 29, no. 5, pp. 2353–2366, 2021.

[14] G. Bianchi, "Performance analysis of the ieee 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, 2000.

[15] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *Journal of the ACM*, vol. 20, no. 1, pp. 46–61, 1973.

[16] K. Cohen, A. Nedic, and R. Srikant, "Distributed learning algorithms for spectrum sharing in spatial random access wireless networks," *IEEE Transactions on Automatic Control*, vol. 62, no. 6, pp. 2854–2869, June 2017.

[17] M. Bahraini, M. Zanon, A. Colombo, and P. Falcone, "Optimal control design for perturbed constrained networked control systems," *IEEE Control Systems Letters*, vol. 5, no. 2, pp. 553–558, 2021.

[18] F. Kserawi, S. Al-Marri, and Q. Malluhi, "Privacy-preserving fog aggregation of smart grid data using dynamic differentially-private data perturbation," *IEEE Access*, vol. 10, pp. 43 159–43 174, 2022.

[19] L. Deng, C.-C. Wang, M. Chen, and S. Zhao, "Timely wireless flows with general traffic patterns: Capacity region and scheduling algorithms," *IEEE/ACM Transactions on Networking*, vol. 25, no. 6, pp. 3473–3486, 2017.

[20] Z. J. Haas and J. Deng, "On optimizing the backoff interval for random access schemes," *IEEE Trans. on Communications*, vol. 51, no. 12, pp. 2081–2090, December 2003.