A Decentralized Medium Access Protocol for Real-Time Wireless Ad Hoc Networks With Unreliable Transmissions

Ping-Chun Hsieh
CESG and Department of ECE
Texas A&M University
College Station, TX 77843, USA
pingchun.hsieh@tamu.edu

I-Hong Hou
CESG and Department of ECE
Texas A&M University
College Station, TX 77843, USA
ihou@tamu.edu

***Abstract*—This paper proposes a feasibility-optimal decentralized algorithm for real-time wireless ad hoc networks, where a strict deadline is imposed for each packet. While centralized scheduling algorithms provide provably optimal theoretical guarantees, they may not be practical in many settings, such as industrial networked control systems. Therefore, it is of great importance to design an algorithm that achieves feasibility optimality in a decentralized manner. To design a decentralized algorithm, we leverage two widely-used functions of wireless devices: carrier sensing and backoff timers. Different from the conventional approach, the proposed algorithm utilizes a collision-free backoff scheme to enforce the transmission priority of different links. This design obviates the capacity loss due to collision with quantifiably small backoff overhead. The algorithm is fully decentralized in the sense that every link only needs to know its own priority, and links contend for priorities only through carrier sensing. We prove that the proposed algorithm is feasibility-optimal. NS-3 simulation results show that the proposed algorithm indeed performs as well as the feasibility-optimal centralized algorithm. Moreover, the results also show that the proposed algorithm converges to optimality very fast.**

I. Introduction

Real-time wireless networks are becoming essential to a variety of existing and emerging applications. For multimedia applications such as virtual reality and live video streaming, video contents need to be delivered from the content providers to the end users within several tens of milliseconds to provide seamless user experience. On the other hand, cyber-physical systems (CPS), such as industrial Internet of Things (IoT) applications and networked transportation systems, usually require ultra-low per-packet latency within several milliseconds as well as very low packet loss rate to achieve reliable real-time control. These characteristics are usually captured by *timely-throughput*, which is defined as the average throughput of on-time packet deliveries.

To achieve the timely-throughput requirements, researchers have been devoting significant efforts to designing scheduling algorithms with provable performance guarantees. In [1], Dua and Bambos study downlink scheduling for packets with deadlines by applying dynamic programming. In [2], Hou et al. propose a mathematical framework for designing centralized online scheduling algorithms for deadline-constrained wireless networks over static unreliable channels. This framework has been extended to various wireless scenarios, such as fading channels [3], multicast traffic [4], broadcast traffic [5], co-existence of real-time and non-real-time traffic [6], arbitrary traffic patterns [7], and wireless ad hoc networks [8, 9]. These algorithms make scheduling decisions based on the state information of each wireless link, such as virtual queue length and the number of packet arrivals. While these scheduling algorithms have provably good performance, they need to maintain state information at the centralized controller (such as a Wi-Fi access point (AP) or a cellular base station) and might not be practical in certain settings. For example, when there are multiple collocated access points sharing the same wireless medium, either for better coverage or higher network density, it is required to incorporate additional coordination between the access points in order to apply these centralized scheduling algorithms. Such coordination can be extremely difficult for real-time wireless networks due to stringent per-packet deadlines. Moreover, there might be direct device-to-device communication without the help of an AP (such as ad hoc mode of IEEE 802.11 and WiFi Direct [10]) for message exchange between sensors and actuators [11]. Furthermore, to apply centralized control in a wireless network with uplink traffic, an access point needs to continually update global information through polling. This may incur prohibitively large scheduling overhead, especially when the network size is large.

In light of the above issues of centralized scheduling, recently researchers have embarked on designing random-access algorithms to allow uncoordinated devices to share the same wireless medium efficiently [12]–[23]. The existing studies on random-access algorithms can be divided into two main categories:

1) **Providing throughput guarantees for *non-real-time* wireless networks**: To achieve throughput-optimality in wireless ad hoc networks, Jiang and Walrand [12] and Rajagopalan, Shah, and Shin [13] propose two different CSMA-based distributed algorithms by using queue-length-based continuous backoff scheme under the assumption of perfect carrier sensing. The results are later extended for utility maximization with congestion control [14, 15], circuit-switched networks [16], and various weight functions for backoff timers [17]. Without assuming perfect carrier sensing, Ni et al. [18] develop a throughput-optimal

queue-length-based CSMA algorithm for wireless ad hoc networks by using discrete backoff timers and control packets. To overcome the limitations of both perfect carrier sensing and control packets, Shah et al. [19] propose a throughput-optimal random access algorithm by intelligently choosing a weight function that depends only on the local queue and sensing information. Lotfinezhad and Marbach [20] propose a CSMA-based algorithm that achieves throughput-optimality as well as order-optimal delay for wireless ad hoc networks, and Paolini et al. [21] propose a coded random access scheme based on slotted ALOHA to support massive uncoordinated wireless access. The above list is by no means exhaustive, but it provides a portrait of the recent progress in random access algorithms for non-real-time wireless networks. Despite the significant progress, none of the above algorithms takes per-packet deadlines into consideration.

2) **Providing timely-throughput guarantees for *real-time* wireless networks**: To accommodate per-packet deadlines, based on a similar approach as [12], Li and Eryilmaz [22] propose the FCSMA algorithm, a CSMA-based distributed implementation of [8] for a fully-connected network. While FCSMA algorithm has been shown to be feasibility-optimal, the optimality results are based on the assumption that there is no capacity loss due to backoff overhead or collisions. For real-time wireless networks with stringent packet deadlines, backoff overhead can lead to significant capacity loss and needs to be taken into account. Moreover, one common issue of CSMA with random backoff is that collision rate increases with the network size. It has been shown analytically that the collision probability for the Distributed Coordination Function (DCF) of Wi-Fi protocols can be prohibitively high, and that the throughput loss due to collision is significant, under the exponential backoff scheme even when the network size is fairly small (e.g. 10 links) [24]. On the other hand, by following a similar approach as [18], Lu et al. [23] present the frame-based CSMA algorithm, which distributedly generates schedules on a per-frame basis by incorporating control packets and a control slot at the beginning of each frame. While the frame-based CSMA algorithm has been shown to be feasibility-optimal for wireless ad hoc networks with reliable transmissions, it is sub-optimal with unreliable channels since the schedules are not adaptive to the packet losses within a frame.

In this paper, we aim to *achieve optimal timely-throughput over unreliable channels in a decentralized manner while avoiding the capacity loss due to channel contention (including collision and backoff overhead).* We are particularly interested in real-time wireless ad hoc networks for industrial control systems, where sensors and actuators exchange critical control messages via multiple APs and direct device-to-device communication. Given the limited distance between devices in the same manufacturing area, each device is likely to cause severe interference to all the other devices. In this regard, we propose decentralized priority-based algorithms for fully-interfering networks to optimize network timely-throughput and address the channel contention issue simultaneously.

Different from conventional CSMA-based algorithms, the proposed algorithm let all the links contend for *transmission priorities* in addition to immediate channel access. Instead of using random backoff, which is commonly used by CSMA-based algorithms, we utilize a collision-free backoff scheme as the tool to determine transmission priorities of all the links. In this way, the proposed algorithm is free from collisions and have quantifiably small backoff overhead. Based on the transmission priorities, the schedules can automatically adapt to packet losses in a fully decentralized fashion. Moreover, by dynamically adjusting the transmission priorities in a randomized manner, the proposed decentralized algorithm achieves optimal timely-throughput as its centralized counterparts.

It is important to note that conventional CSMA-based algorithms usually suffer from slow convergence and large delay due to the "locking" effect [20, 25]. Specifically, CSMA-based algorithms tend to stick to one schedule for a long time and hence result in starvation of the other unscheduled links. To tackle this issue, there have been significant efforts in improving delay performance and convergence time of CSMA algorithms, such as [18, 20, 25]–[31]. By contrast, one advantage of the proposed priority-based algorithm is that the priority structure mitigates the locking effect by nature. Specifically, under any priority ordering, each link receives a non-zero expected timely-throughput depending on its priority index. Therefore, even the link with the lowest priority does not get completely starved. This design shares a similar philosophy as the virtual multi-channel approach proposed by [25]. More details on convergence time will be discussed via simulation in Section VI.

The main contributions can be summarized as follows:

- We propose a generic fully-decentralized priority-based protocol for wireless networks. The proposed protocol requires only carrier sensing and backoff mechanism and does not require any additional control packets or control slots.
- The proposed protocol utilizes a collision-free backoff mechanism and therefore completely avoids capacity loss due to collided transmissions. Meanwhile, the overhead due to backoff is quantifiably small compared to the packet deadlines. The proposed protocol is robust to packet losses resulting from unreliable transmissions.
- For wireless networks with per-packet deadlines, by combining the proposed protocol with virtual queue lengths, we propose a fully-decentralized priority-based algorithm that is feasibility-optimal.
- We evaluate the proposed algorithm via extensive NS-3 simulations and show that it indeed performs as well as the optimal centralized algorithms.

The rest of this paper is organized as follows. Section II describes the system model and notations. Section III discusses a centralized feasibility-optimal algorithm for real-time wireless networks. Section IV presents the proposed decentralized priority-based protocol. In Section V, we apply delivery debt to the proposed protocol and show that it achieves feasibility optimality. Section VI provides simulation results, and Section VII concludes the paper.

II. System Model and Problem Formulation

We study the problem of optimizing timely-throughputs for real-time wireless networks in a decentralized manner. The

network model is described as follows.

A. Network Topology and Transmission Model

We consider a wireless ad hoc network formed by the wireless sensors, actuators, and controllers of a networked control system. To ensure connectivity for all the wireless devices, a networked control system might require multiple APs, each of which serves a subset of these client devices. These wireless entities form a set of N directed links denoted by $\mathcal{N} = \{1, \cdots, N\}$, each of which can serve as a downlink or an uplink between an AP and a client, or a direct communication link between two clients intended for machine-to-machine interactions. For industrial control applications, we may assume that all the nodes in the network are using the same wireless protocol, and hence there is no coexistence issue. Figure 1 shows an example of the wireless network described above.

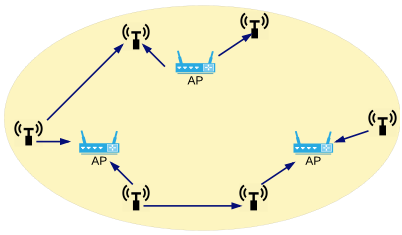

Fig. 1. A network of multiple collocated APs serving multiple clients.

We consider the scenario where all the links share the same frequency band and interfere with each other, i.e. the conflict graph is complete. For industrial control applications, since all the wireless sensors and actuators are located in the same manufacturing area, each link is susceptible to continual interference from the other links. Since all links interfere with each other, if multiple links transmit simultaneously, a transmission collision occurs and all transmissions fail. In practice, to avoid collision due to the hidden terminal problem, the energy level of carrier sensing can be configured to be as low as possible. Moreover, transmissions can also fail due to the unreliable nature of wireless transmissions. Specifically, if link n transmits a packet and the transmission does not suffer from interference, then we assume that the transmission is successful with probability $p_n > 0$. We use $\boldsymbol{p} = [p_n, n \in \mathcal{N}]$ to denote the success probability vector. Note that p_n can be obtained by either probing or learning from the empirical results of past transmissions. Here we simply assume that p_n is known by the transmitter of each link. If a transmission fails, the transmitter might be able to retransmit the same packet, depending on the scheduling algorithm. For simplicity, we also assume that total airtime required for transmitting a single packet (including the airtime of an ACK and the required guard time between transmissions) is the same for all the packets.

To avoid excessive collisions due to interference, *carrier sensing* is one widely-used approach (also known as *listen-before-talk*) to detect transmissions from neighboring wireless links. We assume that all the devices support carrier sensing, i.e. at each time instant each device is able to determine whether the channel is busy. However, each device is not able to overhear transmissions by other devices, since the interference range is usually much larger than the transmission range in wireless communications [32].

B. Packets with Deadlines

To continuously support real-time operations of industrial systems, each link n is associated with a data stream with a strict per-packet relative deadline equal to T time units. Note that the time unit here can be chosen arbitrarily. In many of the related works on wireless scheduling (such as [2, 3, 8]), one unit time is chosen to be the total transmission time of a packet due to the synchronized slot structure embedded in the centralized scheduling algorithms. By contrast, in this paper, we do not impose the slot structure to the proposed decentralized algorithm, so that we can explicitly address the amount of time spent on channel contention. For industrial control applications, the deadline T can be chosen based on the maximum allowable delay bound for control messages or the data sampling period. The packets that are not delivered before their deadlines are dropped. For convenience, we further divide time into *intervals*, each of which has a length of T time units and corresponds to $(kT, (k+1)T]$ for some $k \in \mathbb{N} \cup \{0\}$.

For each link, packets arrive at the beginning of each interval in a probabilistic manner. For each link n, let $A_n(k)$ be a nonnegative integer-valued random variable which denotes the number of arriving packets in the k-th interval and let $\boldsymbol{A}(k) := [A_n(k), n \in \mathcal{N}]$ be the corresponding arrival vector. For each link n and interval k, we assume that $A_n(k)$ is upper bounded by some constant $A_{\max} < \infty$. We model the arrival process $\{\boldsymbol{A}(k), k \in \mathbb{N} \cup \{0\}\}$ as a sequence of temporally independent and identically distributed (i.i.d.) random vectors with mean vector $\boldsymbol{\lambda} = [\lambda_n, n \in \mathcal{N}]$ for all interval k. Note that in each interval, the numbers of packet arrivals of different links might still be correlated. If all the packets are delivered by the end of the interval, all the links stay idle till the beginning of next interval.

In the rest of the paper, we use $(\mathcal{N}, \boldsymbol{A}, T, \boldsymbol{p})$ to denote a wireless network described above.

C. Timely-Throughput and Feasibility Optimality

To guarantee reliable operation of the networked control system, each link n is required to maintain a minimum timely-throughput of q_n packets per interval. Equivalently, each link n requires a *minimum delivery ratio* of $\rho_n := \frac{q_n}{\lambda_n}$. Note that when there is exactly one packet arrival in each interval for each link, timely-throughput is exactly the same as delivery ratio [2]. For networked control systems with high reliability requirements, the delivery ratios are usually chosen to be close to 1.

To achieve the required timely-throughput, we focus on designing transmission policies that determine the actions (transmit or stay idle) of each link. Let $\mathcal{H}_t$ be the history of the network up to time t. $\mathcal{H}_t$ includes all the past packet arrivals, all past actions of each link, and the outcomes of all the past transmissions. We use Π to denote the set of all the history-dependent transmission policies. For any policy $\eta \in \Pi$, it can be either randomized or deterministic, centralized or decentralized.

Now, we formally define the notion of feasibility optimality as follows. Let $S_n(k)$ be the number of delivered packets in the k-th interval, for each link n.

Definition 1. For any $K \in \mathbb{N}$, the *timely-throughput deficiency* up to K-th interval of each link n is defined as $\left(q_n - \frac{\sum_{k=0}^{K-1} S_n(k)}{K}\right)^+$. Moreover, *total timely-throughput deficiency* up to K-th interval is defined as $\sum_{n=1}^{N}\left(q_n - \frac{\sum_{k=0}^{K-1} S_n(k)}{K}\right)^+$.

Note that timely-throughput deficiency reflects the difference between the required timely-throughput and the empirical timely-throughput.

Definition 2. For a network described by $(\mathcal{N}, \boldsymbol{A}, T, \boldsymbol{p})$, a timely-throughput requirement vector $\boldsymbol{q} = [q_n, n \in \mathcal{N}]$ is *fulfilled* under some policy η if total timely-throughput deficiency converges to 0 in probability as $K \to \infty$, where $(\cdot)^+ := \max\{0, \cdot\}$.

Definition 3. For a network described by $(\mathcal{N}, \boldsymbol{A}, T, \boldsymbol{p})$, a timely-throughput requirement vector $\boldsymbol{q} = [q_n, n \in \mathcal{N}]$ is *feasible* if there exists some policy that fulfills $\boldsymbol{q}$. Besides, $\boldsymbol{q}$ is *strictly feasible* if $q_n > 0$ for all n and there exists some $\alpha \in (0,1)$ such that $(1+\alpha)\boldsymbol{q}$ is also feasible.

Definition 4. For a network described by $(\mathcal{N}, \boldsymbol{A}, T, \boldsymbol{p})$, the *feasible region* $\mathcal{Q}$ is defined as the set of all feasible timely-throughput requirement vectors. Similarly, the *strict feasible region* $\mathcal{Q}^*$ is the set of all strict feasible timely-throughput requirement vectors.

In this paper, we focus on the strict feasible region $\mathcal{Q}^*$. Next, we introduce the definition of feasibility optimality.

Definition 5. A policy $\eta \in \Pi$ is *feasibility-optimal* if it fulfills all strictly feasible $\boldsymbol{q} \in \mathcal{Q}^*$.

Regarding the notations, we use boldface letters to denote vectors and matrices. We use $||\cdot||_\infty$ to denote the L^∞-norm of a vector or a matrix. Besides, we use $\mathbb{R}_{\geq 0}$ to denote the set of all nonnegative real numbers and use $\mathbb{P}\{\cdot\}$ to denote the probability of an event.

III. Centralized Feasibility-Optimal Algorithm

In this section, we study a centralized feasibility-optimal algorithm, namely the *extended largest-debt-first* (ELDF) scheduling. To begin with, we first introduce the notions of debt and debt influence functions.

A. Delivery Debt and Debt Influence Functions

To study packet deliveries with deadlines, we consider a virtual queue length, namely *delivery debt*, to capture the amount of on-time packet deliveries of each link. Recall that $S_n(k)$ denotes the number of delivered packets in the k-th interval, for each link n. Since the packets that miss their deadlines are dropped, we have $S_n(k) \leq A_n(k)$, for all n and k. Let $d_n(k)$ be the delivery debt of link n at the beginning of interval k. Then, $d_n(k)$ evolves as follows [3]:

$$d_n(k+1) = d_n(k) - S_n(k) + q_n, \tag{1}$$

with $d_n(0) = 0$, for all n. Equivalently, we have $d_n(k) = kq_n - \sum_{j=0}^{k-1} S_n(j)$. Based on (1), we know $d_n(k)$ reflects the difference between actual timely-throughput and the required timely-throughput.

To design transmission policies based on delivery debt, we consider a class of functions called *debt influence functions*.

Definition 6. A functions $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is said to be a *debt influence function* if

1) f is nondecreasing, continuous, Riemann integrable, and satisfies that $\lim_{x\to\infty} f(x) = \infty$.
2) Given any finite $c \in \mathbb{R}$, f satisfies that $\lim_{x\to\infty} \frac{f(x+c)}{f(x)} = 1$. Equivalently, given any $\epsilon > 0$, there exists a constant $B > 0$ such that $1 - \epsilon \leq \frac{f(x+c)}{f(x)} \leq 1 + \epsilon$, $\forall x \geq B$.

Moreover, we use $\mathcal{F}$ to denote the set of all debt influence functions.

For example, $f(x) = x^m$ with $m \geq 0$ and $f(x) = \log_a x$ with $a > 1$ are valid debt influence functions. On the other hand, $f(x) = a^x$ with $a > 1$ is not a debt influence function.

B. A Sufficient Condition of Feasibility Optimality

We first introduce a sufficient condition which helps us design feasibility-optimal transmission policy. Note that the network can be viewed as a controlled Markov chain with inequality constraints. We then have useful results on feasibility-optimal randomized policies in the following lemma.

Lemma 1. For any strictly feasible vector $\boldsymbol{q} \in \mathcal{Q}^*$, there exists a stationary randomized policy that fulfills $\boldsymbol{q}$ based on a probability distribution that only depends on the number of packets to be delivered and the number of time slots remaining in the current interval.

Proof. This is a direct result of [33]. □

We first introduce a sufficient condition of feasibility optimality in the following lemma.

Lemma 2. For a transmission policy $\eta \in \Pi$, for any debt influence function $f \in \mathcal{F}$, if given any $\delta \in (0,1)$ there exists a constant $B_0 > 0$ such that in every interval we have

$$\mathbb{E}^{\eta}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] \tag{2}$$

$$\geq (1-\delta)\max_{\eta' \in \Pi} \mathbb{E}^{\eta'}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right], \tag{3}$$

whenever $||\boldsymbol{d}(k)||_\infty > B_0$, then for any strictly feasible vector $\mathbf{q} \in \mathcal{Q}^*$, the Markov chain induced by $\{\mathbf{d}(k)\}$ is positive recurrent under policy η, and hence η is feasibility-optimal.

Proof. This can be proved by using Lyapunov drift analysis over one interval. Due to space limitations, the complete proof is provided in Appendix A of the technical report [34]. □

Remark 1. Note that by choosing $f(x) = x$ in Lemma 2, we can recover Theorem 2 in [3], which considers only linear debt influence function. Besides, Lemma 2 serves a similar purpose as Claim 1 in [35]. Different from [35], we consider packets with deadlines in this paper.

C. A Feasibility-Optimal Centralized Scheduling Algorithm

Motivated by the *Largest-Debt-First* (LDF) scheduling policy proposed in [2, 3], we consider the following extended version of LDF (ELDF) as shown in Algorithm 1.

Algorithm 1 Extended Largest-Debt-First Scheduling Policy

1: At the beginning of the k-th interval, sort all of the N clients such that

$$f(d^+_{m_1}(k))p_{m_1} \geq f(d^+_{m_2}(k))p_{m_2} \geq \cdots \geq f(d^+_{m_N}(k))p_{m_N}. \quad (4)$$

In this way, $[m_n, n \in \mathcal{N}]$ determines the transmission priority of each link.

2: Transmit packets based on the priority ordering of (4) till the end of the k-th interval.

Remark 2. Note that ELDF is a priority-based policy, which updates its priority ordering at the beginning of each interval. By choosing $f(x) = x$, the ELDF policy becomes equivalent to the LDF policy, as both of them assign priorities according to $\boldsymbol{d}(k)$.

Next, we show that the ELDF policy is feasibility-optimal.

Lemma 3. Under unreliable channels described in Section II-A, the ELDF policy using delivery debt maximizes $\mathbb{E}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \;\middle|\; \boldsymbol{d}(k)\right]$ among all the policies in Π.

Proof. This is a direct result of Theorem 3 in [3]. □

Based on Lemma 3, we know that ELDF always satisfies the sufficient condition provided in Lemma 2.

Proposition 1. Under unreliable channel model and delivery debt $\{\boldsymbol{d}(k)\}$, the ELDF policy is feasibility-optimal.

Proof. This can be proved by Lemma 2 and Lemma 3. □

IV. Decentralized Priority-Based Protocol

While Section III has introduced the feasibility-optimal centralized algorithm, such an algorithm may be infeasible to implement in a fully decentralized fashion. Specifically, Algorithm 1 requires the complete knowledge of the number of packets generated by each link, as well as the delivery debt of each link. On the other hand, we also note that Algorithm 1 has the nice structure of being a priority-based policy, that is, it assigns a priority to each link at the beginning of each interval. In this section, we propose a generic decentralized priority-based algorithm and describe the features of the proposed algorithm.

A. Transmission Priorities and Permutations

To deal with transmission priorities, we introduce some useful definitions regarding permutations as follows.

Definition 7. For a set of integers $\mathcal{N} = \{1, \cdots, N\}$, a *permutation* $\boldsymbol{\sigma}$ of $\mathcal{N}$ is a bijective map from $\mathcal{N}$ to $\mathcal{N}$.

With a slight abuse of notation, we represent a permutation $\boldsymbol{\sigma}$ in vector form as $\boldsymbol{\sigma} = [\sigma_1, \cdots, \sigma_N]$. Next, we consider the transition from one permutation to another.

Definition 8. A *transposition* of a permutation $\boldsymbol{\sigma}$ is defined as an exchange of two entries in $\boldsymbol{\sigma}$. Moreover, an *adjacent transposition* (i, j) of $\boldsymbol{\sigma}$ is defined as an exchange of two entries σ_i and σ_j with $|\sigma_i - \sigma_j| = 1$, for some $i, j \in \mathcal{N}$.

Definition 9. For any two permutations $\boldsymbol{\sigma}, \boldsymbol{\sigma'} \in \mathfrak{S}_N$, the *symmetric difference* is defined as $\boldsymbol{\sigma} \triangle \boldsymbol{\sigma'} := \{n : \sigma_n \neq \sigma'_n\}$.

Example 1. Let $N = 4$ and consider two permutations: $\boldsymbol{\sigma} = [2, 1, 4, 3]$, $\boldsymbol{\sigma'} = [2, 4, 1, 3]$. By definition, we have $\boldsymbol{\sigma} \triangle \boldsymbol{\sigma'} = \{2, 3\}$, and $\boldsymbol{\sigma'}$ can be obtained by applying an adjacent transposition $(2, 3)$ to $\boldsymbol{\sigma}$.

In the rest of the paper, we use $\mathfrak{S}_N$ to denote the set of all permutations of $\{1, 2, ..., N\}$. In each interval k, let $\boldsymbol{\sigma}(k) = [\sigma_1(k), ..., \sigma_N(k)]$ be the transmission priority vector, where each $\sigma_n(k)$ denotes the priority index of link n, and $\boldsymbol{\sigma}(k)$ can be any permutation in $\mathfrak{S}_N$. The link with the highest priority should have priority index equal to 1.

B. A Generic Decentralized Priority-Based Protocol

In this section, we introduce a generic decentralized priority-based (DP) protocol as shown in Algorithm 2. In the DP protocol, the transmissions are designed to be collision-free by making different links choose different backoff timers in each interval. Specifically, in each interval, each link has a unique priority index within the range $\{1, \cdots, N\}$ and chooses its backoff timer based on this priority, as shown in Step 4. Note that this design is completely decentralized in the sense that each link only needs to know its own priority index.

To dynamically adjust priorities, in each interval, a pair of links with priority difference equal to 1 (denoted by $C(k)$ and $C(k)+1$ in Algorithm 2) is chosen uniformly at random as candidates for swapping priorities. These two links swap priorities only when the link of priority $C(k)$ tend to move down and the link of priority $C(k)+1$ tend to move up. Without using any control message, the above event can be detected by both links via carrier sensing plus proper choices of backoff timers. As shown in Step 3 and 4, each of the two swapping candidates determines its backoff timer by using an individual coin toss with parameter μ_n. As shown in (7), if the link of priority $C(k)$ tends to move down, it increases its priority index by 1 only if the channel is sensed busy when backoff number decreases to 1. The event that the channel is busy when backoff number decreases to 1 indicates that the link of priority $C(k)+1$ also tends to move up. Similarly, as described in (8), if the link of priority $C(k)+1$ tends to move up, it detects that the link of priority $C(k)$ also tends to move down only if the channel is sensed idle when backoff number decreases to 1. Any change in priority index will be enforced in the next interval. By using the above mechanism, the two candidate links achieve implicit coordination completely via carrier sensing. For any non-candidate link, it simply remains at the same priority for one interval.

Algorithm 2 Decentralized Priority (DP) Protocol With Randomized Reordering (Link n in the k-th interval)

1: At the beginning of the k-th interval, select an integer-valued uniform random variable $C(k)$ with value between 1 and $N-1$ by using a random seed shared by all the devices (for example, system time).

2: If $n \in \{C(k), C(k)+1\}$ and link n has no packet arrival in the current interval, then link n generates an empty packet (for the purpose of priority claiming) and puts it in its buffer.

3: Generate a random variable $\xi_n(k)$ locally as

$$\xi_n(k) = \begin{cases} 1 & \text{, with probability } \mu_n \\ -1 & \text{, with probability } 1-\mu_n \end{cases} \tag{5}$$

where $\mu_n \in (0,1)$ is a parameter chosen by link n.

4: Given the priority index $\sigma_n(k-1)$, determine the backoff timer $\beta_n(k)$ for the current interval as

$$\beta_n(k) = \begin{cases} \sigma_n(k-1)-1 & \text{, if } \sigma_n(k-1) < C(k) \\ \sigma_n(k-1)+1 & \text{, if } \sigma_n(k-1) > C(k)+1 \\ \sigma_n(k-1)-\xi_n(k) & \text{, if } \sigma_n(k-1) = C(k) \\ & \text{or } \sigma_n(k-1) = C(k)+1 \end{cases} \tag{6}$$

5: At any time instant, if link n does not hear any transmission from any other link, link n continues on the backoff procedure. If $n = C(k)$, then link n updates its priority index by

$$\sigma_n(k) = \begin{cases} \sigma_n(k-1) & \text{, if } \xi_n(k) = 1 \\ \sigma_n(k-1)+1 & \text{, if } \xi_n(k) = -1 \text{ and channel is} \\ & \text{busy when backoff timer} \\ & \text{decreases to 1} \\ \sigma_n(k-1) & \text{, otherwise} \end{cases} \tag{7}$$

If $n = C(k)+1$, then link n updates its priority index by

$$\sigma_n(k) = \begin{cases} \sigma_n(k-1)-1 & \text{, if } \xi_n(k) = 1 \text{ and channel is} \\ & \text{idle when backoff timer} \\ & \text{decreases to 1} \\ \sigma_n(k-1) & \text{, otherwise} \end{cases} \tag{8}$$

If n is neither $C(k)$ nor $C(k)+1$, then link n simply updates its priority index by $\sigma_n(k) = \sigma_n(k-1)$.

6: When the backoff timer decreases to zero, link n starts transmitting packets till the end of the k-th interval or until there is no packet left in its buffer.

7: At the end of the current interval, flush all the packets in the buffer and updates network states accordingly.

Remark 3. Note that the DP protocol does not have the synchronized slot structure embedded in many centralized scheduling algorithms (such as [2, 3, 8]). In fact, there might be a short period of idle time due to backoff between two packet transmissions. To make sure that every link obtains the same $C(k)$ in Step 1, the DP protocol only requires a common random seed (e.g. through initial coarse time synchronization).

Remark 4. In Step 6 of the DP protocol, if the remaining time of the current interval is less than the transmission time of a packet, link n simply stays idle till the end of the interval. This "gap" is less than one packet transmission time and can be minimized by choosing a proper combination of packet deadline and packet transmission time.

Remark 5. In Step 3 of the DP protocol, each link needs to choose the parameter μ_n. In Section V, we will discuss how to choose μ_n to achieve feasibility optimality.

Remark 6. Note that the DP protocol can be further generalized to the cases of multiple swapping pairs. Specifically, in Step 1 of Algorithm 2, multiple non-consecutive integers are selected among $\{1, \cdots, N-1\}$, and in Step 4 the backoff timers are determined in a similar manner. Please refer to [34] for more detailed discussions.

To make the idea of DP protocol more clear, we consider the following toy example:

Example 2. Consider a network of 4 links $\{1, 2, 3, 4\}$, each of which has channel reliability $p_n = 1$ and exactly 1 packet arrival at the beginning of each interval. Suppose $\boldsymbol{\sigma}(1) = [1, 2, 3, 4]$ and $\boldsymbol{\sigma}(2) = [1, 3, 2, 4]$. As shown in Figure 2, Link 2 and 3 exchange priorities if link 2 and link 3 set backoff timer $\beta_2(k) = 3$ and $\beta_3(k) = 2$ in Step 4 of Algorithm 2, respectively.

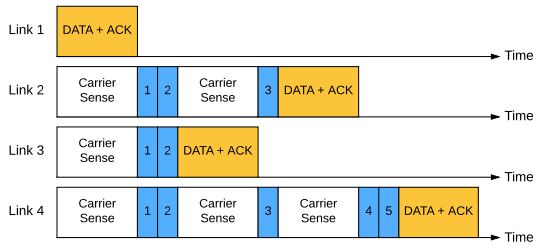

Fig. 2. An example of priority exchange using backoff.

Note that Step 1 to 6 in Algorithm 2 are completely agnostic to packet deadlines. Therefore, Algorithm 2 actually provides a generic approach to implementing priority-based algorithms in a fully-decentralized fashion.

C. Features of the Decentralized Protocol

Before providing theoretical analysis of the decentralized protocol, we first highlight the important features of the protocol.

- **Fully-decentralized**: Under the DP protocol, each link only needs to know its own priority. To update the priority index, each link only needs to determine whether the channel is busy by using carrier sensing. This completely removes the messaging overhead of maintaining network states at an AP required by a centralized scheduling algorithm.
- **Quantifiably small overhead incurred by maintaining priorities**. One advantage of the proposed algorithm is that the overhead can be clearly quantified. The overhead comes from two main parts: (i) *backoff timer*: According to Step 4 of Algorithm 2, we know that the backoff timer of each

link is at most $N+1$. For example, in the widely-used 802.11a/g/n/ac standard, one backoff slot is chosen to be 9 μs, which is usually much smaller than the packet deadline (several milliseconds). Moreover, even shorter backoff slot time is also possible by optimizing carrier sensing functionality and Rx to Tx turnaround time [36]. (ii) *empty packet for claiming priority*: In each interval, there are at most two empty packets, each of which requires much smaller transmission time than that of a typical data packet. For example, in 802.11a protocol, the transmission time of a packet with no payload plus the required interframe spacing is about 70 μs. By contrast, even with the highest data rate of 54 Mbps, the time to send a typical 1500 B data packet plus an ACK is roughly 330 μs [37]. Therefore, the overhead incurred by an empty packet is indeed small.

- **No capacity loss due to collision**. The DP protocol enforces transmission priority by letting each link keep a unique backoff number in each interval. In this way, there is no channel contention issue common to many other CSMA-based algorithms that utilize a random backoff mechanism. Therefore, the proposed protocol completely obviates the capacity loss due to collided transmissions. This is particularly critical for industrial networked control systems, where excessive collisions could happen continually due to massive connectivity.
- **Requires only coarse carrier sensing and initial synchronization**. The DP protocol utilizes the discrete backoff mechanism and therefore requires only coarse carrier sensing (or equivalently, carrier sensing can be non-instantaneous). In order to synchronize packet arrivals as other CSMA-based algorithms (such as [18, 22, 23]), the proposed algorithm assumes initial time synchronization, which can be easily achieved by having APs broadcast messages or simply using GPS time information if available. Therefore, the proposed protocol can be easily implemented in the wireless devices without any additional hardware.
- **No control packets or control slots required**. Different from the Q-CSMA-like algorithms [18, 23], there is no control packet required in the proposed DP protocol. This design greatly reduces the messaging overhead as well as implementation efforts.
- **Robust to packet losses**. Note that under the DP protocol, transmission priorities are maintained by backoff timers and transmission attempts (regardless of the outcome), not control packets. Therefore, the DP protocol is robust to packet losses due to unreliable channels.

D. Analysis of Stationary Distribution

In this section, we study the behavior of the proposed decentralized protocol in steady state. Note that the stochastic process $\{\boldsymbol{\sigma}(k), k \in \mathbb{N}_0\}$ can be modeled as a discrete-time Markov chain with a finite state space $\mathfrak{S}_N$ and a stationary $N! \times N!$ transition probability matrix $\boldsymbol{X} = [X_{\boldsymbol{\sigma},\widehat{\boldsymbol{\sigma}}}, \boldsymbol{\sigma}, \widehat{\boldsymbol{\sigma}} \in \mathfrak{S}_N]$. Let $\boldsymbol{\sigma}, \widehat{\boldsymbol{\sigma}}$ be two permutations in $\mathfrak{S}_N$. If $\boldsymbol{\sigma} \triangle \widehat{\boldsymbol{\sigma}} = \{i, j\}$ for some $i, j \in \mathcal{N}$ and (i, j) forms an adjacent transposition, we have

$$\boldsymbol{X}_{\boldsymbol{\sigma},\widehat{\boldsymbol{\sigma}}} = \frac{(1-\mu_i)\mu_j}{N-1} \cdot \mathbb{P}\{R_i + R_j \geq 1\}, \tag{9}$$

where R_i and R_j denote the number of transmissions (including empty packets) made by link i and j in the current interval, respectively. Otherwise, $\boldsymbol{X}_{\boldsymbol{\sigma},\widehat{\boldsymbol{\sigma}}} = 0$. In (9), the term $1/(N-1)$ comes from randomization (Step 1 of Algorithm 2), the term $(1-\mu_i)\mu_j$ represents the probability of the event that the candidate links tend to exchange priority indices (Step 3 and 4 of Algorithm 2), and $\mathbb{P}\{R_i + R_j \geq 1\}$ represents the probability of the event that at least one of the candidate links needs to transmit before confirming the exchange of priority indices (Step 5 of Algorithm 2). To ensure that $\boldsymbol{X}_{\boldsymbol{\sigma},\widehat{\boldsymbol{\sigma}}}$ in (9) is nonzero, we impose a mild technical condition as follows:

- (C1) With a non-zero probability, the total number of packet arrivals in one interval is less than the number of available transmission attempts in the interval.

Next, we present an important property of the Markov chain $\{\boldsymbol{\sigma}(k)\}$ as below.

Lemma 4. Under Algorithm 2 with condition (C1), the Markov chain $\{\boldsymbol{\sigma}(k)\}$ is irreducible and aperiodic.

Proof. Due to space limitations, the complete proof is provided in Appendix B of the technical report [34]. □

Based on Lemma 4, we are able to derive the stationary distribution of the Markov chain $\{\boldsymbol{\sigma}(k)\}$ under Algorithm 2.

Proposition 2. Under Algorithm 2 with condition (C1), the stationary Markov chain $\{\boldsymbol{\sigma}(k)\}$ is time-reversible and has the unique stationary distribution π^* as

$$\pi^*(\boldsymbol{\sigma}) = \frac{\prod_{n=1}^{N} \left(\frac{\mu_n}{1-\mu_n}\right)^{g(\sigma_n)}}{Z}, \quad \forall \boldsymbol{\sigma} \in \mathfrak{S}_N \tag{10}$$

where

$$Z := \sum_{\boldsymbol{\sigma} \in \mathfrak{S}_N} \prod_{n=1}^{N} \left(\frac{\mu_n}{1-\mu_n}\right)^{g(\sigma_n)} \tag{11}$$

$$g(j) := \begin{cases} N-j, & \text{if } 1 \leq j \leq N \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

Proof. To show that $\{\boldsymbol{\sigma}(k)\}$ is time-reversible, we simply verify that (10)-(12) satisfy the detailed balance equations [38]. For any two transmission priority vectors $\boldsymbol{\sigma} = (\sigma_1, \cdots, \sigma_N)$ and $\widetilde{\boldsymbol{\sigma}} = (\widetilde{\sigma}_1, \cdots, \widetilde{\sigma}_N)$, if the symmetric difference $\boldsymbol{\sigma} \triangle \widetilde{\boldsymbol{\sigma}} = \{i, j\}$ and $\sigma_i = \widetilde{\sigma}_j = m$ and $\sigma_j = \widetilde{\sigma}_i = m+1$, then we have

$$\frac{\pi^*(\boldsymbol{\sigma})}{\pi^*(\widetilde{\boldsymbol{\sigma}})} = \frac{\prod_{n=1}^{N} \left(\frac{\mu_n}{1-\mu_n}\right)^{g(\sigma_n)}}{\prod_{n=1}^{N} \left(\frac{\mu_n}{1-\mu_n}\right)^{g(\widetilde{\sigma}_n)}} = \frac{\frac{\mu_i}{1-\mu_i}}{\frac{\mu_j}{1-\mu_j}}. \tag{13}$$

Since $\boldsymbol{\sigma}$ and $\widetilde{\boldsymbol{\sigma}}$ share the same transmission ordering for priority 1 through $m-1$, then $\mathbb{P}\{R_i + R_j \geq 1\}$ is the same under $\boldsymbol{\sigma}$ and $\widetilde{\boldsymbol{\sigma}}$. By (9) and (13), we know that $\pi^*(\boldsymbol{\sigma}) \cdot \boldsymbol{X}_{\boldsymbol{\sigma},\widetilde{\boldsymbol{\sigma}}} = \pi^*(\widetilde{\boldsymbol{\sigma}}) \cdot \boldsymbol{X}_{\widetilde{\boldsymbol{\sigma}},\boldsymbol{\sigma}}$. By Lemma 4, we know $\{\boldsymbol{\sigma}(k)\}$ is irreducible and aperiodic, and therefore the stationary distribution is unique. □

V. Decentralized Priority Algorithm For Feasibility Optimality

In this section, we elaborate on how to achieve feasibility optimality by choosing proper parameters for the randomized reordering in Step 3 of the DP protocol.

A. A Decentralized Priority Algorithm Using Delivery Debt

In the ELDF policy, transmission priorities are determined by the product of the value of debt influence function and the channel reliability. Based on this observation, we shall choose μ_n as a function of delivery debt $d_n(k)$ such that μ_n increases with $d_n(k)$. Motivated by the Glauber dynamics [13], we choose μ_n for Step 3 in Algorithm 2 as

$$\mu_n(k) = \frac{\exp(f(d_n^+(k))p_n)}{R + \exp(f(d_n^+(k))p_n)}, \tag{14}$$

where f is the debt influence function and R is a predetermined positive constant. Note that similar forms to (14) have also been adopted in [15, 18, 22]. Meanwhile, in order to apply the results of Lemma 4 and Proposition 2, the Markov chain $\{\boldsymbol{\sigma}(k)\}$ is required to be stationary. Here we consider the concept of *two-time-scale separation* for the Markov chain $\{\boldsymbol{\sigma}(k)\}$. Specifically, we choose μ_n to be a slowly-changing function of $d_n(k)$ such that the evolution of μ_n is much slower than the evolution of $\{\boldsymbol{\sigma}(k)\}$. For example, [13] conjectures that $f(x) = \log\log x$ achieves sufficiently slow evolution of the underlying Markov chain. This result is later relaxed to $f(x) \approx \log(x)$ by [17] for better convergence time. Besides, [18] also shows via simulation that $f(x) = \log x$ achieves the best delay performance. In this way, we are able to approximate the Markov chain $\{\boldsymbol{\sigma}(k)\}$ by its "quasi-stationary" characteristics. This argument has been used extensively in the existing literature for both continuous-time and discrete time Markov chains, such as [13, 15, 18, 39, 40]. Therefore, we rely on this assumption to show feasibility optimality of the proposed algorithm for brevity.

In the rest of the paper, we will refer to Algorithm 2 with $\mu_n(k)$ in (14) as the *debt-based decentralized priority algorithm* (DB-DP). Next, we summarize the important properties of the Markov chain $\{\boldsymbol{d}(k)\}$ under the DB-DP algorithm.

Proposition 3. Under the DB-DP algorithm, the Markov chain $\{\boldsymbol{\sigma}(k)\}$ is irreducible and aperiodic. Moreover, the Markov chain $\{\boldsymbol{\sigma}(k)\}$ is reversible and has the unique stationary distribution π^* as

$$\pi^*(\boldsymbol{\sigma}, k) = \frac{\exp\left(\sum_{n=1}^N g(\sigma_n) f(d_n^+(k))p_n\right)}{Z(\boldsymbol{d}(k))}, \quad \forall \boldsymbol{\sigma} \in \mathfrak{S}_N \tag{15}$$

where

$$Z(\boldsymbol{d}(k)) := \sum_{\boldsymbol{\sigma}\in\mathfrak{S}_N} \exp\left(\sum_{n=1}^N g(\sigma_n) f(d_n^+(k))p_n\right) \tag{16}$$

$$g(j) := \begin{cases} N-j, & \text{if } 1 \le j \le N \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

Proof. Under two-time-scale separation, (15)-(17) can be shown by directly substituting (14) for μ_n in (10)-(12). □

B. Proof of Feasibility Optimality

In this section, we show that the DB-DP algorithm indeed achieves feasibility optimality. Recall that in Section IV-C, we explicitly quantify the overhead incurred by maintaining priorities. For ease of exposition, in this section, we assume that the time of each backoff slot and the time to transmit an empty packet are zero. We introduce the following definition:

Definition 10. The proposed decentralized priority-based protocol is said to be *idealized* if the time of each backoff slot and the time to transmit an empty packet are both zero.

Accordingly, we let the packet deadline to be as long as T packet transmissions. In Section VI, we will provide more detailed discussion on how to determine packet deadlines, packet transmission time, and backoff slots, etc.

Proposition 4. Under the idealized DB-DP algorithm (denoted by η), for any $\delta \in (0,1)$, there exists a $B > 0$ such that in every interval we have

$$\mathbb{E}^{\eta}\left[\sum_{n=1}^N f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] \tag{18}$$

$$\ge (1-\delta)\max_{\eta'\in\Pi} \mathbb{E}^{\eta'}\left[\sum_{n=1}^N f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right], \tag{19}$$

whenever $||\boldsymbol{d}(k)||_\infty > B$.

Proof. We show (18)-(19) by using the stationary distribution obtained by Proposition 2. Different from the ELDF policy, the DB-DP algorithm does not maintain the "optimal" transmission ordering of (4) with probability 1. In spite of this, we can still show that the transmission priority ordering under the DB-DP algorithm is close to "optimal" with high probability when delivery debts are sufficiently large. Due to space limitations, we include the complete proof in Appendix C of the technical report [34]. □

Remark 7. The main challenge of proving Proposition 4 is two-fold: (i) We need to calculate (18)-(19) based on transmission priority ordering instead of the exact schedules. It is not immediately clear how to compare different priority orderings in terms of (18). This is an essential difference between our proof and conventional drift analysis for MaxWeight-type scheduling policies. (ii) The transmission priority ordering is stochastic. Therefore, we need to consider all possible priority orderings.

Theorem 1. The idealized DB-DP algorithm is feasibility-optimal.

Proof. This can be proved by Lemma 2 and Proposition 4. □

VI. Simulation Results

In this section, we evaluate the proposed algorithm via extensive NS-3 simulations. NS-3 provides a discrete-event environment, which allows us to implement all the features of the DB-DP algorithm. The source code for the simulations can be found at [41]. We are particularly interested in two applications: (i) real-time video delivery and (ii) ultra-low-latency control information delivery. While these two applications both require deadline guarantees, they greatly differ in traffic pattern as well as the required level of wireless service.

Throughout the simulations, we consider IEEE 802.11a as the underlying MAC protocol with physical data rate equal to 54 Mbps. Under 802.11a, a backoff slot is set to be 9 μs to account for non-instantaneous carrier sensing. We compare the proposed DB-DP algorithm with the LDF policy (a special case of ELDF with $f(x) = x$) and the FCSMA algorithm proposed by [22]. For the DB-DP algorithm, we choose the debt influence function as $f(x) = \log(\max\{1, 100(x+1)\})$ and $R = 10$. For the FCSMA algorithm, we consider its discretized version with the same parameters as suggested in [22]. We evaluate the performance of the three algorithms based on the *total timely-throughput deficiency* defined in Definition 1. By Definition 2, we know that under a given timely-throughput requirement vector $\mathbf{q}$, $\mathbf{q}$ is fulfilled if and only if the total timely-throughput deficiency converges to zero as simulation time goes to infinity. Note that in the following simulations, we might observe a small non-zero total timely-throughput deficiency even for feasible $\mathbf{q}$ due to the finite simulation time.

A. Real-Time Video Delivery

In this section, we provide simulation results for real-time video delivery, which is required by many industrial networked control systems for machine vision and process surveillance [42]. We assume the payload size of each data packet is 1500 B and the packet deadline is 20 ms. The total airtime required by sending a data packet plus an ACK and the interframe spacing is about 330 μs. Under LDF, there are up to 60 transmissions in each interval. On the other hand, under the proposed DB-DP algorithm, there might be 1 or 2 fewer transmissions in each interval due to the time spent on backoff slots and empty packets for claiming priority. To capture the bursty traffic pattern of video streams, we assume that the number of packet arrivals at each link n in each interval is uniformly distributed within $\{1, 2, 3, 4, 5, 6\}$ with probability α_n and is 0 with probability $1-\alpha_n$. Therefore, the arrival rate $\lambda_n = 3.5\alpha_n$, for each link n. The simulation time is 5000 intervals, or equivalently 100 seconds.

First, we consider a fully-symmetric network of 20 links, i.e. all the links have the same channel reliability $p_n = p*$, arrival rate λ_n, and delivery ratio ρ_n. Accordingly, we let $\alpha_n = \alpha^*$, for all n. We assume that $p* = 0.7$ and the delivery ratio $\rho_n = 0.9$. Figure 3 shows the total timely-throughput deficiency under three algorithms. Note that the proposed DB-DP algorithm achieves almost the same performance as LDF, which is known to be a feasibility-optimal centralized algorithm. On the other hand, FCSMA supports only about 70% of the maximum admissible α^* (about 0.62 based on the results of LDF in Figure 3). This is mainly due to the capacity loss resulting from significant backoff overhead as well as collided transmissions as discussed in Section I. Next, we study the achievable delivery ratio under a fixed arrival rate. We assume $\alpha^* = 0.55$, which is slightly smaller than the maximum admissible α^* suggested by Figure 3. Figure 4 shows the total timely-throughput deficiency under $\alpha^* = 0.55$ and various delivery ratios. Similar to Figure 3, we observe that DB-DP and LDF achieve almost the same level of delivery ratio while FCSMA suffers from significant loss of timely-throughput.

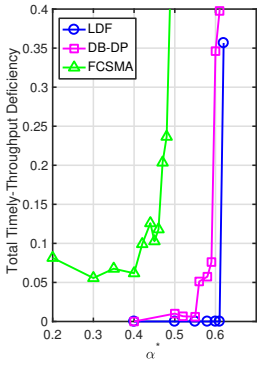

Fig. 3. Total timely-throughput deficiency of the symmetric network under 90% delivery ratio.

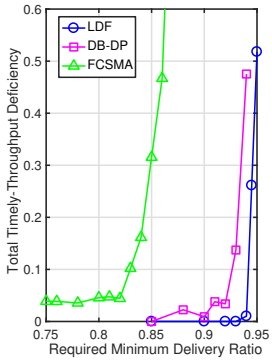

Fig. 4. Total timely-throughput deficiency of the symmetric network under a fixed arrival rate with $\alpha^* = 0.55$.

To show the convergence time under DB-DP and LDF, we consider the timely-throughput of the link with the lowest priority at time 0 under $\alpha^* = 0.55$ and 93% delivery ratio, as shown in Figure 5. As expected, LDF indeed achieves a relatively small convergence time due to the nature of centralized control. We also observe that DB-DP achieves a convergence time (e.g. within 1% neighborhood of the timely-throughput requirement) comparable to that of LDF policy. As mentioned in Section I, the DB-DP algorithm naturally alleviates the starvation problem of conventional CSMA algorithms due to the design of priority structure. To make this more clear, Figure 6 shows the average timely-throughput of each link under a *fixed priority ordering* and $\alpha^* = 0.6$. As expected, the average timely-throughput increases with priority (with small variations due to random arrivals) and the link with the lowest priority (priority index = 20) still receives non-zero timely-throughput. Further results on convergence time can be found in the technical report [34].

Next, we consider an asymmetric scenario by dividing the 20 links into two groups of equal size:

- Group 1: each link has $p_n = 0.5$ and $\alpha_n = 0.5\alpha^*$.
- Group 2: each link has $p_n = 0.8$ and $\alpha_n = \alpha^*$.

The required delivery ratio is still 0.9 for each link. Figure 7 and Figure 8 show the group-wide total timely-throughput deficiency under a fixed delivery ratio 90% and a fixed $\alpha^* = 0.7$, respectively. Similar to Figure 3 and 4, DB-DP algorithm still achieves almost the same performance as the LDF policy with the existence of network asymmetry. Note that under FCSMA, group 1 suffers from much larger deficiency than group 2. This is mainly due to the discretization design of [22], where the range of delivery debt is divided into a finite

number of sections and each section is mapped to one of the predetermined sizes of the contention window. Therefore, the size of contention window is the same for any delivery debt above a certain threshold. Hence, FCSMA becomes completely oblivious when delivery debt is sufficiently large and fails to respond to the changes in delivery debt.

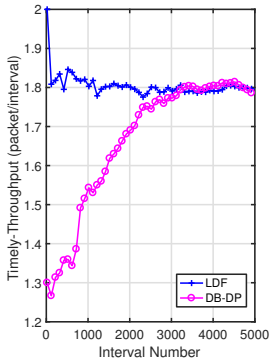

Fig. 5. Comparison of convergence time under DB-DP and LDF policy with $\alpha^* = 0.55$ and 93% delivery ratio.

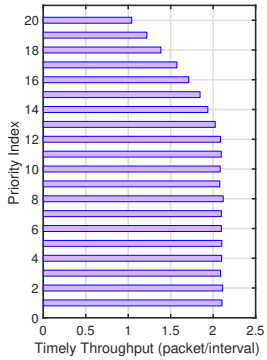

Fig. 6. Average timely-throughput under a fixed priority ordering, $\alpha^* = 0.6$.

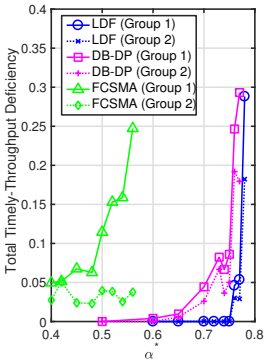

Fig. 7. Group-wide total timely-throughput deficiency of asymmetric network under 90% delivery ratio.

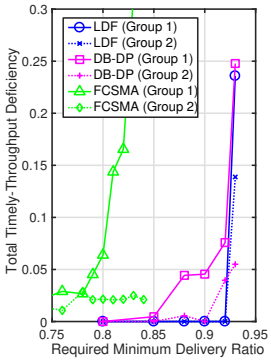

Fig. 8. Group-wide total timely-throughput deficiency of asymmetric network under $\alpha^* = 0.7$.

B. Ultra-Low-Latency Control Information Delivery

In this section, we present simulation results for ultra-low-latency information delivery, which is critical to industrial networked control systems. In order for the machines to reliably perform mission-critical tasks, the machines and the controllers need to continuously exchange time-critical control messages, which require a per-packet deadline below 10 ms [43, 44]. These control packets are usually small (less than 100 B) but require much more stringent per-packet deadlines than other data packets. Here we assume the per-packet deadline T is 2 ms and the payload size of each packet is 100 B. Under the same MAC-layer settings, the total airtime to transmit one packet plus an ACK is roughly 120 μs. Regarding the arrival process, we assume that the numbers of packet arrivals of each link in each interval form a sequence of i.i.d. Bernoulli random variables with mean λ^*. The total simulation time is 20000 intervals, or equivalently 40 seconds.

For ease of exposition, we consider a fully-symmetric network of 10 links, where each link has the same $p_n = 0.7$, $\lambda_n = \lambda^*$, and the delivery ratio equal to 0.99. Figure 9 and Figure 10 show the total timely-throughput deficiency under a fixed delivery ratio 99% and under a fixed $\lambda^* = 0.78$, respectively. Note that under LDF there are 16 available transmissions in each interval. On the other hand, DB-DP might have 1 or 2 fewer transmissions in one interval due to the overhead of backoff slots and empty packets. In spite of this, DB-DP still achieves a timely-throughput close to that of the LDF policy when the packet deadline is as low as 2 ms.

Based on the discussion in Section VI-A and VI-B, we see that the DB-DP indeed achieves the same level of real-time wireless service as the LDF policy.

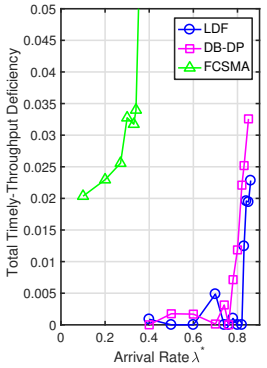

Fig. 9. Total timely-throughput deficiency under 99% delivery ratio.

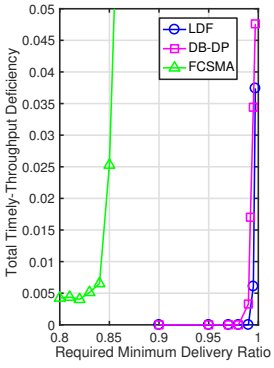

Fig. 10. Total timely-throughput deficiency under a fixed $\lambda^* = 0.78$.

VII. Concluding Remarks

This paper proposes a feasibility-optimal decentralized algorithm for real-time wireless ad hoc networks over unreliable wireless channels. We first present a generic decentralized priority-based protocol that utilizes only carrier sensing and collision-free backoff mechanism. Under the proposed protocol, the overhead of maintaining transmission priority ordering is small and can be easily quantified. Next, we combine the generic decentralized protocol with delivery debt and show that it is feasibility-optimal. We evaluate the performance of the proposed decentralized algorithm via extensive NS-3 simulations and show that it performs as well as the feasibility-optimal centralized algorithm.

Acknowledgement

The authors would like to thank Prof. Eytan Modiano at Massachusetts Institute of Technology for the helpful discussions. This material is based upon work supported in part by NSF under contract number CNS-1719384, the US Army Research Laboratory and the US Army Research Office under contract/Grant Number W911NF-15-1-0279, Office of Naval Research under Contract N00014-18-1-2048, and NPRP Grant 8-1531-2- 651 of Qatar National Research Fund (a member of Qatar Foundation).

REFERENCES

[1] A. Dua and N. Bambos, "Downlink wireless packet scheduling with deadlines," *IEEE Transactions on Mobile Computing*, vol. 6, no. 12, 2007.

[2] I. H. Hou, V. Borkar, and P. R. Kumar, "A Theory of QoS for Wireless," in *Proc. of INFOCOM*, 2009, pp. 486–494.

[3] I.-H. Hou, "Scheduling heterogeneous real-time traffic over fading wireless channels," *IEEE/ACM Transactions on Networking (TON)*, vol. 22, no. 5, pp. 1631–1644, 2014.

[4] K. S. Kim, C.-P. Li, and E. Modiano, "Scheduling multicast traffic with deadlines in wireless networks," in *Proc. of INFOCOM*, 2014, pp. 2193–2201.

[5] I.-H. Hou, "Broadcasting delay-constrained traffic over unreliable wireless links with network coding," *IEEE/ACM Transactions on Networking (TON)*, vol. 23, no. 3, pp. 728–740, 2015.

[6] J. J. Jaramillo and R. Srikant, "Optimal scheduling for fair resource allocation in ad hoc networks with elastic and inelastic traffic," in *Proc. of INFOCOM*. IEEE, 2010, pp. 1–9.

[7] 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 (TON)*, 2017.

[8] J. J. Jaramillo, R. Srikant, and L. Ying, "Scheduling for optimal rate allocation in ad hoc networks with heterogeneous delay constraints," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 5, pp. 979–987, 2011.

[9] X. Kang, W. Wang, J. J. Jaramillo, and L. Ying, "On the performance of largest-deficit-first for scheduling real-time traffic in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 24, no. 1, pp. 72–84, 2016.

[10] D. Camps-Mur, A. Garcia-Saavedra, and P. Serrano, "Device-to-device communications with Wi-Fi Direct: overview and experimentation," *IEEE Wireless Communications*, vol. 20, no. 3, pp. 96–104, 2013.

[11] J. Chen, X. Cao, P. Cheng, Y. Xiao, and Y. Sun, "Distributed collaborative control for industrial automation with wireless sensor and actuator networks," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 12, pp. 4219–4230, 2010.

[12] L. Jiang and J. Walrand, "Convergence and stability of a distributed csma algorithm for maximal network throughput," in *Proc. Conference on Decision and Control (CDC)*, 2009, pp. 4840–4845.

[13] S. Rajagopalan, D. Shah, and J. Shin, "Network adiabatic theorem: an efficient randomized protocol for contention resolution," in *Proc. of ACM SIGMETRICS performance evaluation review*, vol. 37, no. 1, 2009, pp. 133–144.

[14] L. Jiang and J. Walrand, "A distributed CSMA algorithm for throughput and utility maximization in wireless networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 18, no. 3, pp. 960–972, 2010.

[15] L. Jiang, D. Shah, J. Shin, and J. Walrand, "Distributed random access algorithm: scheduling and congestion control," *IEEE Transactions on Information Theory*, vol. 56, no. 12, pp. 6182–6207, 2010.

[16] D. Shah, J. Shin *et al.*, "Randomized scheduling algorithm for queueing networks," *The Annals of Applied Probability*, vol. 22, no. 1, pp. 128–171, 2012.

[17] J. Ghaderi and R. Srikant, "On the design of efficient CSMA algorithms for wireless networks," in *Proc. of Conference on Decision and Control (CDC)*. IEEE, 2010, pp. 954–959.

[18] J. Ni, B. Tan, and R. Srikant, "Q-CSMA: Queue-length-based CSMA/CA algorithms for achieving maximum throughput and low delay in wireless networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 3, pp. 825–836, 2012.

[19] D. Shah, J. Shin, and P. Tetali, "Medium access using queues," in *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*. IEEE, 2011, pp. 698–707.

[20] M. Lotfinezhad and P. Marbach, "Throughput-optimal random access with order-optimal delay," in *Proc. of INFOCOM*. IEEE, 2011, pp. 2867–2875.

[21] E. Paolini, C. Stefanovic, G. Liva, and P. Popovski, "Coded random access: applying codes on graphs to design random access protocols," *IEEE Communications Magazine*, vol. 53, no. 6, pp. 144–150, 2015.

[22] B. Li and A. Eryilmaz, "Optimal distributed scheduling under time-varying conditions: A fast-csma algorithm with applications," *IEEE Transactions on Wireless Communications*, vol. 12, no. 7, pp. 3278–3288, 2013.

[23] N. Lu, B. Li, R. Srikant, and L. Ying, "Optimal distributed scheduling of real-time traffic with hard deadlines," in *Proc. of Conference on Decision and Control (CDC)*, 2016, pp. 4408–4413.

[24] 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.

[25] P.-K. Huang and X. Lin, "Improving the delay performance of CSMA algorithms: A virtual multi-channel approach," in *Proc. of INFOCOM*. IEEE, 2013, pp. 2598–2606.

[26] C.-H. Lee, S.-Y. Yun, Y. Yi *et al.*, "From Glauber dynamics to Metropolis algorithm: Smaller delay in optimal CSMA," in *Proc. of ISIT*. IEEE, 2012, pp. 2681–2685.

[27] K.-K. Lam, C.-K. Chau, M. Chen, and S.-C. Liew, "Mixing time and temporal starvation of general CSMA networks with multiple frequency agility," in *Proc. of ISIT*. IEEE, 2012, pp. 2676–2680.

[28] D. Shah, N. David, and J. N. Tsitsiklis, "Hardness of low delay network scheduling," *IEEE Transactions on Information Theory*, vol. 57, no. 12, pp. 7810–7817, 2011.

[29] D. Shah and J. Shin, "Delay optimal queue-based CSMA," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 38, no. 1. ACM, 2010, pp. 373–374.

[30] L. Jiang, M. Leconte, J. Ni, R. Srikant, and J. Walrand, "Fast mixing of parallel Glauber dynamics and low-delay CSMA scheduling," *IEEE Transactions on Information Theory*, vol. 58, no. 10, pp. 6541–6555, 2012.

[31] V. G. Subramanian and M. Alanyali, "Delay performance of CSMA in networks with bounded degree conflict graphs," in *Proc. of ISIT*. IEEE, 2011, pp. 2373–2377.

[32] K. Xu, M. Gerla, and S. Bae, "How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks," in *Proc. of GLOBECOM*, vol. 1, 2002, pp. 72–76.

[33] F. J. Beutler and K. W. Ross, "Optimal policies for controlled Markov chains with a constraint," *Journal of mathematical analysis and applications*, vol. 112, no. 1, pp. 236–252, 1985.

[34] P.-C. Hsieh and I.-H. Hou, "A decentralized medium access protocol for real-time wireless ad hoc networks with unreliable transmissions," Tech. Rep., February 2018. [Online]. Available: http://people.tamu.edu/%7Elleyfede/decentralized.pdf

[35] A. Eryilmaz, R. Srikant, and J. R. Perkins, "Stable scheduling policies for fading wireless channels," *IEEE/ACM Transactions on Networking (TON)*, vol. 13, no. 2, pp. 411–424, 2005.

[36] E. Magistretti, K. K. Chintalapudi, B. Radunovic, and R. Ramjee, "Wifi-nano: reclaiming wifi efficiency through 800 ns slots," in *Proc. of International Conference on Mobile computing and networking*. ACM, 2011, pp. 37–48.

[37] IEEE 802 Standard Working Group, "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications: high-speed physical layer in the 5GHz Band," 1999.

[38] H. Chen and D. D. Yao, *Fundamentals of queueing networks: Performance, asymptotics, and optimization*. Springer Science & Business Media, 2001, vol. 46.

[39] D. N. C. Tse, R. G. Gallager, and J. N. Tsitsiklis, "Statistical multiplexing of multiple time-scale Markov streams," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 6, pp. 1028–1038, 1995.

[40] G. G. Yin and Q. Zhang, *Discrete-time Markov chains: two-time-scale methods and applications*. Springer, 2006, vol. 55.

[41] https://github.com/pinghsieh/ns3-sim.

[42] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Computer networks*, vol. 51, no. 4, pp. 921–960, 2007.

[43] A. Frotzscher, U. Wetzker, M. Bauer, M. Rentschler, M. Beyer, S. Elspass, and H. Klessig, "Requirements and current solutions of wireless communication in industrial automation," in *Proc. of IEEE International Conference on Communications Workshops (ICC)*, 2014, pp. 67–72.

[44] J. Åkerberg, M. Gidlund, and M. Björkman, "Future research challenges in wireless sensor and actuator networks targeting industrial automation," in *Proc. of IEEE International Conference on Industrial Informatics (INDIN)*, 2011, pp. 410–415.