A High-order Markov Chain Based Scheduling Algorithm for Low Delay in CSMA Networks

Jaewook Kwak, Chul-Ho Lee and Do Young Eun

***Abstract*—Recently, several CSMA algorithms based on the Glauber dynamics model have been proposed for multihop wireless scheduling, as viable solutions to achieve the throughput optimality, yet simple to implement. However, their delay performance still remains unsatisfactory, mainly due to the nature of the underlying Markov chains that imposes a fundamental constraint on how the link state can evolve over time. In this paper, we propose a new approach toward better queueing delay performance, based on our observation that the algorithm needs not be Markovian, as long as it can be implemented in a distributed manner. Our approach hinges upon utilizing past state information observed by local link and then constructing a high-order Markov chain for the evolution of the feasible link schedules. We show in theory and simulation that our proposed algorithm, named *delayed CSMA*, achieves the throughput optimality, and also provides much better delay performance by effectively 'de-correlating' the link state process (and thus resolves link starvation). Our extensive simulations demonstrate that the delay under our algorithm can be often reduced by a factor of 20 over a wide range of scenarios, compared to the standard Glauber-dynamics-based CSMA algorithm.**

I. Introduction

Medium access control (MAC), which decides link level data transmission, is a central component in wireless packet scheduling. As the MAC plays an important role in achieving efficient channel utilization and providing quality of service for diverse wireless applications, designing an efficient MAC algorithm has been considered to be of significant importance. In a rich and long history of research on this subject, the most commonly believed goal is to achieve the following properties: (i) high throughput, (ii) low delay, and (iii) simple and distributed implementation. These three criteria, however, have different tradeoffs among them, and hence developing an algorithm that has all the properties simultaneously is still a challenging problem.

In wireless networks, the property of high throughput is often determined by the packet arrival rate region under which the algorithm stabilizes the network queues. A classical algorithm, the max-weight scheduling (MWS) [1], is known to achieve the largest rate region under a general independent set constraint model. The MWS, however, is not deemed practical since it requires global information to solve a complicated combinatorial optimization problem in each time instance. Many heuristics such as greedy-maximal scheduling (GMS) or maximal-matching algorithms are considered as alternatives to MWS, but they may achieve only a fraction of the capacity region [2], [3], or are throughput-optimal only on certain types of network topology [4], [5], [6]. There are also several methodologies for achieving the throughput optimality in general network [7], [8], [9], but they have turned out to incur excessive message passing in many cases.

A great advance has recently been made toward this problem in a class of CSMA scheduling, where the throughput optimality can be achieved in a simple and distributed manner, e.g. [10], [11], [12], [13], [14]. The basic idea is based on the so-called Glauber dynamics, which is a Monte Carlo Markov Chain (MCMC) method that often provides an approximate solution to a combinatorial optimization problem. The key enabler in realizing the method for the CSMA scheduling is to achieve a desired probability distribution for the max-weight schedule by locally controlling the CSMA parameters without explicit knowledge of arrival rate or neighboring information. For example, in [10], a distributed algorithm is developed to adaptively choose the CSMA parameter with locally observable information, and the throughput optimality is shown under the time-scale separation assumption (the system converges to its stationary regime quickly enough before adaptation of the CSMA parameters). In another approach [12], the optimality is established without the assumption by taking the parameters as slowly varying function of the queue-size. More general sufficient conditions on the function for the throughput optimality have been studied in [14].

Although the fact that these CSMA algorithms guarantee the throughput optimality is an appealing merit, simulations often demonstrate that they incur large backlogs on the queue, and the resulting delay performance is far from being satisfactory. Thus motivated, in this paper, we mainly focus on improving the (queueing) delay performance without sacrificing the other properties, i.e., high throughput and simple implementation.

A. Related work

In the literature, there have been several attempts to improve the delay performance. For example, in [15], Shah and Shin adopt a coloring operation to the CSMA algorithm so as to achieve a constant-bounded property on the average delay. And, Lotfinezhad and Marbach [16] propose an algorithm called U-CSMA that attempts to resolve link starvation problem by periodically resetting the ongoing schedules to an empty schedule. In these approaches, however, it is assumed that the networks have a polynomial growth structure [15],

The authors are with Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27695-7911, Email: {jkwak, clee4, dyeun}@ncsu.edu. This work was supported in part by National Science Foundation under Grant No. CNS-1217341.

978-1-4799-3360-0/14/$31.00

or the topologies are of torus or grid types [16]. It is thus questionable if such an improvement still holds for *any* arbitrary network topology. In [17], Lam et al. show that the delay performance can be improved when multiple channels are available, but such multi-channels may not be present in practice. Huang and Lin [18] propose a virtual multiple channel scheme to maximize the aggregate of utilities of links, while achieving low delay. However, adapting their scheme for stabilizing network queues fed by different feasible, yet unknown, arrival rates may be a non-trivial task, and also their scheme requires an additional signaling among neighbors, which is certainly a new overhead. In [19], Lee et al. propose a suit of generalized versions of Glauber dynamics that all achieve the same stationary distribution and suggest that Metropolis-Hastings algorithm leads to better delay performance, but its improvement is still below the performance gain obtained by our approach, as will be shown later.

B. Our contributions

In this paper, we propose a new approach toward better queueing delay performance, based on our observation that the algorithm needs not be Markovian, as long as it can be implemented in a distributed manner. Our algorithm, termed *delayed CSMA*, updates the next schedule not based on the current status, but on 'several steps back' past state information, thus necessitating high-order Markov chain modeling. This schedule update based on 'delayed' information, somewhat counter-intuitively, provides a significant gain in delay performance by effectively removing the strong correlations that persist in the link state process and thus alleviating link starvation problem. It is also implementable in a completely distributed fashion, without any additional message overhead.

In particular, we prove that our algorithm achieves the throughput optimality, yet provides much better delay performance in the steady state by 'reshaping' the correlation structure to our advantage, while keeping the stationary distribution of the schedules intact. Our extensive simulations show that the delay under our algorithm is smaller than the conventional CSMA algorithm based on the Glauber dynamics by often a factor of 20 over a wide range of scenarios. Our analysis also offers an interesting viewpoint about the role of the mixing time on the delay performance [11], [20] by showing the tradeoff between faster mixing time in the transient phase and smaller correlations in the steady state. In addition, since the main idea behind our algorithm is to decide the next schedule depending on several-steps-back state information, thereby leading to low delay, we expect that our approach can be similarly invoked to improve the delay performance of other scheduling algorithms (modeled by reversible Markov chains) updating the next schedule based on the current status.

II. Preliminaries

A. Network model

We consider a wireless network with a *conflict graph* $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ where $\mathcal{N}$ is the set of links (transmitter-receiver pair), and $\mathcal{E}$ is the set of edges which represents conflict relationship between links. An edge $(i,j) \in \mathcal{E}$ exists between two links i and j if simultaneous use of the two leads to failure of communications. We define a schedule by $\boldsymbol{\sigma} = (\sigma_v)_{v\in\mathcal{N}} \in \{0,1\}^{|\mathcal{N}|}$, which represents the set of transmitting links. A link v (or node v in the conflict graph $\mathcal{G}$) is active if it is included in the schedule, i.e., $\sigma_v = 1$, and is inactive if otherwise. A *feasible* schedule is a set of links that can be active at the same time slot according to the conflict relationship $\mathcal{E}$. Thus, a feasible schedule $\boldsymbol{\sigma}$ should satisfy the independent set constraint i.e., $\sigma_i + \sigma_j \le 1$ for all $(i,j) \in \mathcal{E}$. We denote by Ω the set of all feasible schedules.

In our model, each link is associated with a queue fed by some exogenous traffic arrivals and serviced when the link is active. We consider that a packet arrives to the queue of link v at each time slot t according to a Bernoulli process $A_v(t)$, i.e., $A_v(t)$, $t = 1, 2, \ldots$ are $i.i.d.$ with $\mathbb{E}\{A_v(t)\} = \eta_v$. Let $\boldsymbol{\eta} = (\eta_v)_{v\in\mathcal{N}}$ be the set of arrival rates to the queues in the network. Let $Q(t) = (Q_v(t))_{v\in\mathcal{N}}$ be the number of packets in the queue at time t. Then the queue dynamics is governed by the following recursion:

$$Q_v(t) = [Q_v(t-1) + A_v(t) - \sigma_v(t)]^+, \quad t \ge 1, \qquad (1)$$

where $[x]^+ = \max\{0, x\}$. See Figure 1 for illustration.

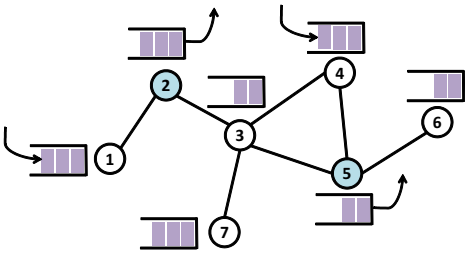


Fig. 1. An instance of schedule where links 2 and 5 are active in a conflict graph with $|\mathcal{N}| = 7$.

B. CSMA scheduling as a Glauber dynamics

The basic idea of throughput-optimal CSMA is to utilize the Glauber dynamics as a link scheduling algorithm. Direct adaptation of the traditional Glauber dynamics to scheduling is as follows. For a graph $\mathcal{G}$, at every time $t \in \mathbb{N}$, a link v is chosen uniformly at random from $\mathcal{N}$. Then,

$$\text{If } \sum_{w\in N_v} \sigma_w(t-1) = 0, \text{ then } \begin{cases} \sigma_v(t) = 1, & \text{w.p. } \frac{\lambda_v}{1+\lambda_v}, \\ \sigma_v(t) = 0, & \text{w.p. } \frac{1}{1+\lambda_v}, \end{cases}$$

$$\text{otherwise, } \sigma_v(t) = 0,$$

$$\text{and for all } w \ne v, \text{ set } \sigma_w(t) = \sigma_w(t-1),$$

where $N_v = \{w : (v,w) \in \mathcal{E}\}$ is a set of neighboring nodes of v, and λ_v is a parameter called *fugacity*. Given $\lambda_v > 0$ for all $v \in \mathcal{N}$, the schedule $\boldsymbol{\sigma}(t)$ forms a Markov chain which is irreducible, aperiodic, and reversible over Ω, and achieves the stationary distribution given by $\pi(\boldsymbol{\sigma}) = \frac{1}{Z}\prod_{i\in\mathcal{N}} \lambda_i^{\sigma_i}$ where $Z = \sum_{\boldsymbol{\sigma}\in\Omega}\prod_{i\in\mathcal{N}} \lambda_i^{\sigma_i}$ is a normalizing constant.

A practical CSMA algorithm uses a modified version of the above procedure. First, the fugacity λ_v is set to be changing dynamically over time t. It is typically chosen as a function of

locally observable $Q_v(t)$ information. Second, multiple links are allowed to be selected in a single time slot [21], [11]. In this procedure, a set of links that do not conflict with each other is selected. This can be achieved in a distributed fashion through a simple randomized procedure. For instance, each link attempts to access the channel with access probability a_v, $v \in \mathcal{N}$, and link v is then selected with probability

$$m_v = a_v \prod_{j \in N_v} (1 - a_j). \tag{2}$$

The set of chosen links $D(t)$ from this procedure is called a *decision schedule* at time t. More practical implementation tailored to IEEE 802.11 can be found in [21].

It is not difficult to find a continuous time version of the above model. However, the discrete time model has several advantages over the continuous one. For example, by synchronizing the time slots, the well known hidden and exposed terminal problems can be effectively resolved [21]. Also, the continuous time model often requires ideal channel sensing mechanism that may not be feasible in practice [22]. The discrete time model has thus been widely used in the literature, e.g., [11], [14], [22], [21], [18]. We also consider the discrete time model throughout the paper.

C. Capacity region and stability

The capacity region of the network is the set of all arrival rates $\boldsymbol{\eta}$ for which there exists a scheduling algorithm that can support the arrivals. It is known [1] that the region is given by the convex combination of all feasible schedules, i.e.,

$$\mathbb{C} = \left\{ \sum_{\boldsymbol{\sigma} \in \Omega} \theta_{\boldsymbol{\sigma}} \boldsymbol{\sigma} : \sum_{\boldsymbol{\sigma} \in \Omega} \theta_{\boldsymbol{\sigma}} = 1, \theta_{\boldsymbol{\sigma}} \geq 0, \forall \boldsymbol{\sigma} \in \Omega \right\}$$

We first collect several definitions. Let $W_v(t)$ be a weight function associated with a link $v \in \mathcal{N}$ at time slot t. It was shown in [1] that MWS is throughput-optimal with $W_v(t) = Q_v(t)$ (see below for definition in more general settings), provided that it can select a maximum-weight schedule $\boldsymbol{\sigma}^*(t)$ in every time slot where

$$\boldsymbol{\sigma}^*(t) = \arg\max_{\boldsymbol{\sigma} \in \Omega} \sum_{v \in \boldsymbol{\sigma}} W_v(t).$$

This has been generalized in [23] as follows. For all $v \in \mathcal{N}$, set link weights as $W_v(t) = h(Q_v(t))$ for some monotone increasing functions $h : [0, \infty) \to [0, \infty)$. (See [23] for precise definitions for the weight functions $h(\cdot)$.) In this case, a scheduling algorithm is said to be *throughput-optimal* if for all arrival rates inside the capacity region, network queues are stable in the sense that

$$\limsup_{K \to \infty} \frac{1}{K} \sum_{t=0}^{K-1} \mathbb{E} \left[\left(\sum_{v \in \mathcal{N}} h^2(Q_v(t)) \right)^{1/2} \right] < \infty. \tag{3}$$

Under some mild assumptions, the authors in [12], [13] have shown that with the choice of $\lambda_v(t) = e^{W_v(t)}$ where the weights $W_v(t)$ are in the form of $\log\log(Q_v(t))$, the conventional CSMA algorithm via Glauber dynamics as in Section II-B achieves the throughput optimality. In [14], the choice of $\log\log(\cdot)$ for $h(\cdot)$ has been slightly generalized into $\log(\cdot)/g(\cdot)$ for a function g that increases arbitrarily slowly. When the whole system including all the queue-lengths is a Markov chain, the condition in (3) implies that the chain is positive recurrent [24], [14], [13]. It is however, worth mentioning that the condition in (3) itself is established under a fairly general case in that the system of $Q_v(t)$ doesn't need to be a Markov chain.

III. CSMA Scheduling Based on High-order Markov Chain

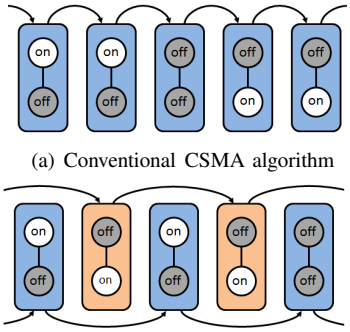
A. Motivation and our approach

According to the standard queueing theory, the queueing delay is governed by not only the long-term average arrival and service rates, but also their higher-order statistics such as their correlations or dependency over time [25], [26]. Indeed, there are a number of works in the literature suggesting that positive correlations have an adverse impact on the queueing delay [27], [28], [29], [30]. With this in mind, in this paper, we aim at developing a new, distributed algorithm similar to the current CSMA-based ones [12], [14], [15], [21], [13], [19], but offering far superior delay performance by effectively reducing such correlations in any general network topology, while keeping the throughput optimality intact.

Our motivation comes from the fact that the service process $\sigma_v(t)$ at link v under the standard CSMA policy is often heavily correlated over time. (More detailed discussion is in Section III-B.) This is because once CSMA finds a schedule, it tends to stay in its similar set of schedules for a long time [16]. To illustrate, consider for example two links that interfere with each other, so that only one link can be active at a time. At a particular moment, if a schedule of the two links is 'active-inactive', the inactive link first has to wait until the active link releases the channel occupation. In this case, transition to next possible state is limited to 'active-inactive' or 'inactive-inactive' state, and direct transition to 'inactive-active' state is impossible. (See Figure 2(a).) This phenomenon hinders frequent switch between schedules, leading to starvation for the corresponding inactive link.

The method we propose here effectively resolves this problem. The main idea is as follows. Suppose we have two schedulers that respectively generate schedules independently, while preserving the feasibility constraint for each time slot. If we choose to use one scheduler at every odd time index, and the other one at every even time index (see Figure 2(b)), it is now possible to make a transition from 'active-inactive' directly to 'inactive-active' state, which would be impossible under the conventional CSMA. This alternate use of different schedulers produces more drastic change of states in consecutive time slots, thereby alleviating link starvation while maintaining the same long-term frequency of being active.

Given the potential benefits, we generalize this idea to the use of multiple schedulers in a round-robin manner. Throughout the paper, we will use a notation T to indicate the number of such schedulers. In practice, this can be easily implemented in a distributed setting by having all links together update



(a) Conventional CSMA algorithm

(b) Proposed approach: Delayed CSMA

Fig. 2. Comparison between the conventional CSMA and our proposed approach. A box indicates a schedule, and arrows indicate state transitions.

their schedules based on T-step-back state. For this purpose, each link only needs to remember its last T channel states. This way, the whole system behaves *as if* there are T separate schedulers (or chains) taking turns to generate next schedules. Building upon this idea and applying to the CSMA scheduling, we next present our proposed algorithm, named *delayed CSMA*. Note that if $T = 1$, our algorithm reduces to the conventional CSMA-based scheduling algorithm.

Algorithm 1 Delayed CSMA

Initialize: for all links $i \in \mathcal{N}$, $\sigma_i(t) = 0$, $t = 0, 1, \ldots, T-1$.
At each time $t \geq T$: links find a decision schedule, $\mathcal{D}(t)$ through a randomized procedure, and
for all links $i \in \mathcal{D}(t)$ **do**
  if $\sum_{j \in N_i} \sigma_j(t-T) = 0$ **then**
    $\sigma_i(t) = 1$ with probability $\frac{\lambda_i}{1+\lambda_i}$
    $\sigma_i(t) = 0$ with probability $\frac{1}{1+\lambda_i}$
  else
    $\sigma_i(t) = 0$
  end if
end for
for all links $j \notin \mathcal{D}(t)$ **do**
  $\sigma_i(t) = \sigma_i(t-T)$
end for

B. Understanding correlation structure of the standard CSMA

Before proceed to explain the details about our algorithm, we first study how much correlations are present in the service process $\sigma_v(t)$ for the queue at each link, induced by the standard Glauber dynamics.

To set the stage, consider a homogeneous Markov chain $\boldsymbol{\sigma}(t) \in \Omega$ denoting a feasible configuration by the Glauber dynamics at time slot t, assuming fugacity parameter λ_v is set to be a constant. Since we are interested in the long-run behavior of queueing performance, without loss of generality, we can assume that the Markov chain $\boldsymbol{\sigma}(t)$ is in its stationary regime, i.e., $\mathbb{P}\{\boldsymbol{\sigma}(t) = \boldsymbol{\sigma}\} = \pi(\boldsymbol{\sigma})$, for all $t \geq 0$. We write $\pi(B_v) \triangleq \sum_{\boldsymbol{\sigma} \in B_v} \pi(\boldsymbol{\sigma}) = \mathbb{P}_\pi\{\sigma_v(t) = 1\}$ for the long-term proportion of service availability at link v, where $B_v \triangleq \{\boldsymbol{\sigma} \in \Omega : \sigma_v = 1\} \subseteq \Omega$ is a set of all feasible schedules for which v is active. For the rest of the paper, we mostly focus on the service process and queueing dynamics at a given link $v \in \mathcal{N}$, so we will drop the subscript v and write $\sigma(t)$ for $\sigma_v(t)$ (similarly η for η_v, $A(t)$ for $A_v(t)$, and $Q(t)$ for $Q_v(t)$) unless otherwise necessary. Let $\psi(k) = \mathrm{Cov}\{\sigma(t), \sigma(t+k)\}/\mathrm{Var}\{\sigma(t)\}$ be the correlation coefficient of lag k for the service process. Then, we have the following proposition that characterizes the correlation of the service process. All proofs (of Propositions 1–5) in this paper are omitted due to the page limit, and they can be found in our technical report [31].

Proposition 1. *Let $q_v = \sum_{\boldsymbol{\sigma}: \sigma_j = 0, \forall j \in N_v} \pi(\boldsymbol{\sigma})$ be the probability that none of neighboring links of v is active. Then,*

$$\psi(1) = 1 - \frac{m_v}{1 + (1-q_v)\lambda_v}, \tag{4}$$

$$\psi(2k) \geq \left(1 - \frac{m_v(2-m_v)}{1+(1-q_v)\lambda_v}\right)^k, \quad k = 1, 2, \ldots. \tag{5}$$

where m_v is the probability that link v is selected in a decision schedule as in (2).

Note that the obtained lower bounds are all positive since $0 \leq m_v \leq 1$. The lower bound is, in general, valid only for even lags, however, it has been shown in [32] that for any finite state reversible Markov chain, the sum of adjacent pairs of correlations, $\psi(2n) + \psi(2n+1)$, is positive, decreasing, and convex in $n \geq 1$. Clearly, the standard Glauber dynamics is a reversible Markov chain on Ω. This implies that, in conjunction with $\psi(1)$ being positive, the negative correlations in $\sigma(t)$, even if they exist, will be of much less magnitude and have lesser impact than positive ones. For instance, Figure 3(a) shows measured correlations $\psi(k)$ for link $v = 3$ in the network topology given in Figure 1, based on the standard Glauber dynamics with access probability $a_i = 0.25$ and fugacity $\lambda_i = 1$ for all $i \in \mathcal{N}$, along with the predicted lower bounds in (4) and (5). We observe that the degree of correlations in the service process is significant and the lower bound in Proposition 1 is in fact quite tight over a wide range of lags.

C. High-order Markov chain model

The key feature of our algorithm over the conventional one is that a new schedule is generated not from the current schedule but from T-step-back past schedule, and in doing so, each link just needs to make a decision based on its own T-step-back channel state. Since every link operates this way, the independent set constraint is satisfied all the time. From an analytical point of view, however, this means that the evolution of schedule $\boldsymbol{\sigma}(t)$ is no longer a Markov chain on Ω, rendering all the results associated with Markov chains inapplicable.

Notwithstanding being non-Markov, our algorithm can still be modeled by a high-order Markov chain of order T on the same state space Ω, as follows:

$$\begin{aligned}&\mathbb{P}\{X_t = x \mid X_{t-1} = x_{t-1}, \ldots, X_0 = x_0\}\\ &= \mathbb{P}\{X_t = x \mid X_{t-1} = x_{t-1}, \ldots, X_{t-T} = x_{t-T}\},\end{aligned}$$

implying the current state depends upon T past history.[1] Our algorithm can then be written as

[1] Alternatively, one can also augment the state space into a product space $\Omega \times \cdots \times \Omega$ (T times) on which $\{X_{t-T+1}, \ldots, X_{t-1}, X_t\}$ becomes a Markov chain, but this would lead to largely intractable descriptions.

$$\mathbb{P}\{X_t = y \mid X_{t-T} = x\} = P(x,y), \tag{6}$$

where $P(x,y)$, $x,y \in \Omega$ is the transition probability of the conventional Markov chain from state x to y.

We here collect several notations and simple facts that hold for finite state Markov chains [33], [34], [35] and will prove useful throughout the analysis. Consider a finite-state, ergodic Markov chain Y_n with its transition matrix P. For functions $f, g : \Omega \to \mathbb{R}$, we define

$$\langle f, \mathbf{P}^{(k)} g \rangle_\mu \triangleq \sum_{x,y\in\Omega} f(x)g(y)\mu(x)P^{(k)}(x,y) = \mathbb{E}_\mu\{f(Y_0)g(Y_k)\},$$

where μ is a probability distribution of Y_0 on Ω, and $P^{(k)}(x,y)$ is k-step transition probability of the chain Y_n. For simplicity, we write $\langle f \rangle_\mu \triangleq \langle f, 1\rangle_\mu = \mathbb{E}_\mu\{f(Y_0)\}$. Also define

$$(\mathbf{P}^{(k)} f)(x) \triangleq \sum_{y\in\Omega} P^{(k)}(x,y) f(x) = \mathbb{E}\{f(Y_k)|Y_0 = x\}. \tag{7}$$

For a (high-order) Markov chain X_t with order T as given in (6), if we define $Y_n^m = X_{nT+m}$ for $0 \le m \le T-1$ and $n = 0, 1, 2, \ldots$, then $\{Y_n^m\}_{n\ge 0}$ for each m is a conventional Markov chain with initial state $Y_0^m = X_m$. Since the chain P is ergodic, we have, for any initial state x,

$$\lim_{k\to\infty} (\mathbf{P}^{(k)} f)(x) = \lim_{k\to\infty} \mathbb{E}\{f(Y_k^m)|Y_0^m = x\} = \langle f \rangle_\pi, \tag{8}$$

where π is the stationary distribution of the chain P. Since this holds for any given $m = 0, 1, \ldots, T-1$, it follows that

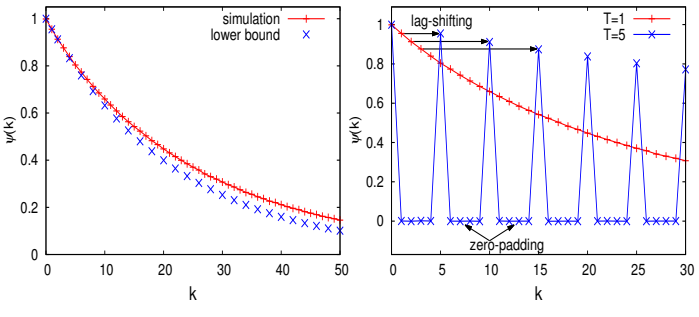
$$\lim_{t\to\infty} \mathbb{E}\{f(X_t)|X_0 = x_1, \ldots, X_{T-1} = x_{T-1}\} = \langle f \rangle_\pi,$$

i.e., the marginal distribution of X_t in the steady-state remains the same and does not change with T. Thus, our delayed CSMA algorithm achieves the same stationary distribution as the standard CSMA algorithm. However, as we describe in the following propositions, different T leads to strikingly different behavior in the second order statistics.

Proposition 2. *(Asymptotic zero-correlation padding) Let X_t be a high-order Markov chain of order T where the transition kernel is given by (6). For any initial distribution, if $k \neq lT$, where $l \in \mathbb{N}$, then $\lim_{t\to\infty} \mathbb{E}\{f(X_t)g(X_{t+k})\} = \langle f \rangle_\pi \langle g \rangle_\pi$, assuming that the expectations exist.*

Proposition 3. *(Asymptotic correlation-lag shifting) Let X_t be a Markov chain of order T with its transition kernel given by (6). For any initial distribution and for any given $k \in \mathbb{N}$, we have $\lim_{t\to\infty} \mathbb{E}\{f(X_t)g(X_{t+kT})\} = \langle f, \mathbf{P}^{(k)} g \rangle_\pi$, assuming that the expectations exist.*

Recall that $\sigma_v(t) = \mathbf{1}\{\boldsymbol{\sigma}(t) \in B_v\}$. Since $\boldsymbol{\sigma}(t)$ under our algorithm is a Markov chain of order T (i.e., modeled as X_t here), Propositions 2 and 3 tell us that the correlations $\psi(k)$ under our delayed CSMA algorithm with order T are first shifted to T times larger lags, and then all padded with zero in-between. To see this effect numerically, we have run the simulations with the same step as in Figure 3(a), but now with $T = 5$. As seen in Figure 3(b), the correlations $\psi(1)$, $\psi(2)$, and $\psi(3)$ for $T = 1$ case respectively gets shifted to $\psi(5)$, $\psi(10)$, and $\psi(15)$ for $T = 5$, and $\psi(k) = 0$ for all $k \not\equiv 0 \pmod 5$. For the rest of the paper, we call these phenomena *zero-padding* and *lag-shifting*, respectively.



(a) (b)

Fig. 3. (a): Correlations of $\sigma_3(t)$ for the network in Figure 1 from simulations and the lower bounds. (b): Two properties of the algorithm: correlation-lag shifting and zero-correlation padding.

IV. Delay and Throughput Analysis

A. Delay performance

In this section, we investigate the efficiency of our delayed-CSMA algorithm of order T in terms of its delay performance. As discussed earlier, our algorithm provides different second-order behavior while preserving the same long-term average statistics. Our analysis hinges upon the common belief that less variability in the service process generally leads to smaller queue-length and delay when the average statistics remains the same. To this end, we assume for now that the fugacity for each link is a fixed constant. This assumption will be relaxed later in Section IV-B, in which we show our algorithm under any finite T is also throughput optimal when the fugacity can be time-varying, set to be some function of the queue-length at each link. At this point, we focus on the long-term behavior of the delay, as any impact of transient phase will eventually fade away as time goes on, and in the standard queueing literature, the behavior of queue-length or delay is discussed mostly in the steady-state. We will briefly touch upon the impact of transient phase under our algorithm later in Section IV-C.

First, Little's law asserts that the average delay is determined by the average queue length given that arrival rate is kept the same. For this reason, we are here interested in the stationary behavior of the queue-length driven by the recursion in (1). Let $I(t) = A(t) - \sigma(t)$ be the *net input* to the queue at time t, with $\mathbb{E}\{I(t)\} = \eta - \pi(B_v) = -\xi < 0$ for the stability of each queue [36]. Let $\mathbf{A}_t = \sum_{k=1}^t A(k)$ and $\mathbf{S}_t = \sum_{k=1}^t \sigma(k)$ be the cumulative amounts of arrival and service over t slots, respectively. For a constant C such that $C > \xi = \pi(B_v) - \eta > 0$, we define $Z(t) \triangleq A(t) - \sigma(t) + C$, and $\mathbf{Z}_t \triangleq \sum_{k=1}^t Z(k)$. Then, the recursion (1) can be rewritten as $Q(t) = [Q(t-1) + Z(t) - C]^+$, i.e., the queue-length evolves as if the arrival process is $Z(t)$ with $\mathbb{E}\{Z(t)\} = C - \xi$ and the service rate is constant C. Thus, the queue-length Q in the steady-state admits the following.

$$\mathbb{P}\{Q > x\} = \mathbb{P}\left\{\sup_{t\ge 0} [\mathbf{Z}_t - Ct] > x\right\}.$$

Our next step is to note that, by similar conditioning on the initial values, we can generalize Propositions 2 and 3 into the case of multiple random variables, say, $f(X_{t+t_1}), g(X_{t+t_2})$, $h(X_{t+t_3}), \ldots$ whose time indices are all distinct in modulo T. Since the choice of functions is arbitrary, this implies that T distinct sub-processes defined by $\sigma^{(i)}(n) = \{\sigma(nT+i)\}_{n\geq 0}$, $i = 1, \ldots, T$ are all independent in the steady state, and the entire correlation structure of the original process $\sigma(n)$ carries over to each of the sub-processes $\sigma^{(i)}(n)$. In particular, let $Z^{(i)}(n) = Z(nT+i)$, $n \geq 0$, for each $i = 1, 2, \ldots, T$. From the zero-padding and lag-shifting properties, and since $A(t)$ is $i.i.d.$ over time t and also independent of $\sigma(t)$, it follows that $\{Z^{(i)}(n)\}_{n\geq 0}$, $i = 1, 2, \ldots, T$ are $i.i.d.$ processes, each of which has $\mathbb{E}\{Z^{(i)}(n)\} = C - \xi$ and $\mathrm{Cov}\{Z^{(i)}(n), Z^{(i)}(n+k)\} = \mathrm{Cov}\{Z(n), Z(n+k)\}$ in the steady state. If we consider a cumulative net-input process up to $t = nT$ for some $n \in \mathbb{N}$, then we can decompose $\mathbf{Z}_t$ into $\mathbf{Z}_t = \sum_{i=1}^{T} \mathbf{Z_t^{(i)}}$ where

$$\begin{aligned}\mathbf{Z_t^{(i)}} \triangleq \sum_{k=0}^{n-1} Z^{(i)}(k) &= \mathbf{A_t^{(i)}} - \mathbf{S_t^{(i)}} + nC \\ &= \sum_{k=0}^{n-1} \left[A(kT+i) - \sigma(kT+i) + C\right].\end{aligned}$$

Thus, the cumulative 'modified net-input' process $\mathbf{Z}_t$ to the queue with constant service rate of C is nothing but a superposition of T $i.i.d.$ replicas of the original ($\mathbf{Z}_t = \sum_{i=1}^{T} \mathbf{Z_t^{(i)}}$), but with its timescale stretched by T-fold. Our algorithm with parameter T thus effectively de-correlates the original service process (or the modified net-input process), regardless of how much correlations persist in the original case caused by any arbitrary topological constraint under the Glauber dynamics. See Figure 4 for illustration. Also,

$$\mathbb{P}\left\{\sup_{t\geq 0}\left[\mathbf{Z}_t - Ct\right] > x\right\} = \mathbb{P}\left\{\sup_{t\geq 0}\sum_{i=1}^{T}\left[\mathbf{Z_t^{(i)}} - (C/T)t\right] > x\right\}.$$

In other words, the queue at link v behaves as if we aggregate T $i.i.d.$ input and also aggregate T different service capacities of C/T each. Thus, we expect that the usual benefits of statistical multiplexing gain and the economies of scales [30], [37], [38], or the principle of "Sharing resources is always better than partitioning." [39] apply here. To quantitatively capture such a gain out of de-correlating the original process under our algorithm, we can also employ the usual Gaussian approximation for $\mathbf{Z}_t$, now a sum of T $i.i.d.$ processes, by appealing to the Central Limit Theorem. Note that $\mathbb{E}\{\mathbf{Z}_t\} = (C-\xi)t$ and we define by $v(t,T)$ the variance of $\mathbf{Z}_t$ under our algorithm with order parameter T. Then, it is known that the queue-length distribution with Gaussian input can be well approximated by [40], [41], [25]

$$\mathbb{P}\{Q > x\} \approx \exp\left(-\inf_{t>0}\frac{(\xi t + x)^2}{2v(t,T)}\right)$$

for a wide range of $x > 0$. We then have the following.

Proposition 4. *Suppose the correlations of $\sigma(t)$ under the original CSMA Glauber dynamics are all non-negative, i.e., $\psi(k) \geq 0$ for all $k \in \mathbb{N}$. Then, for any given $t > 0$, $v(t,T)$ is decreasing (non-increasing) in T.*

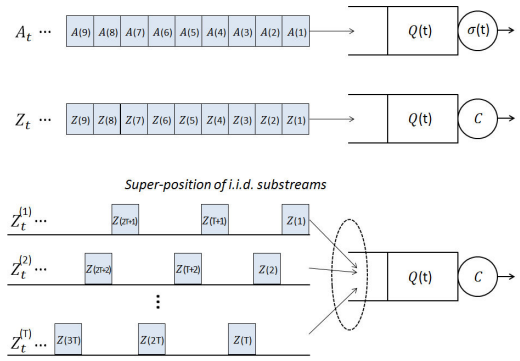


Fig. 4. Our algorithm has an effect of dividing the original net-input process of a single stream (top) into a sum of $i.i.d.$ substreams (bottom) in the steady state.

We expect that positive temporal correlations in the link service process under the original Glauber dynamics prevail for most cases of network topologies and are in accordance with wide-spread link starvation problem in CSMA scheduling. Recall that we have shown in Proposition 1 that $\psi(k)$ is lower bounded by a well defined positive function for every even k. If we consider a lazy chain for the standard Glauber dynamics, $\psi(k) \geq 0$ for all k trivially holds. Even in this case, our high-order chain approach is beneficial to reduce the correlations via zero-padding and lag-shifting, leading to the reduction in the variance of the cumulative net-input to the queue, which subsequently results in smaller queue-length.

B. Throughput optimality

In our delayed CSMA algorithm, we consider $\lambda_v(t) = e^{W_v(t)}$ for the weight function of link v, as discussed in Section II-C. Although the link schedules under our algorithm are updated based on their T-step-back states, we set the weight function $W_v(t)$ to be a function of the current queue length $Q_v(t)$ at time t, rather than T steps ago, such that the system can react more quickly by adjusting the fugacities with the latest information.

With the time-varying fugacities, the state transition matrix becomes time-inhomogeneous, which we write as P_t, a function of $\lambda_v(t), v \in \mathcal{N}$ at time t. For each given such P_t, let π_t be its unique stationary distribution (in a row vector form), i.e., $\pi_t = \pi_t P_t$, and μ_t be the actual distribution of the link schedules $\boldsymbol{\sigma}(t)$ at time t under our algorithm with order parameter T. Then, we have

$$\mu_t = \mu_{t-T} P_{t-1}. \tag{9}$$

Similar to the steps via 'network adiabatic' theorem in [12], [13], a key step in proving the throughput optimality of our algorithm is to show that $\mu_t \approx \pi_t$ for sufficiently large queue lengths under (9). The distance between two probability distributions is characterized by the notion of total variation (TV) distance [35], [34], defined as $\|\nu - \mu\|_{\mathrm{TV}} = \frac{1}{2}\sum_{\sigma\in\Omega}|\nu(\sigma) - \mu(\sigma)|$. Our approach toward the throughput optimality of our proposed algorithm is similar to those in [12], [14], [13], with a little modifications. First, suppose P_t is changing very slowly over time t for all large queue-lengths. (In fact, this can be

achieved by setting the weight function $W_v(t)$ as a very slowly increasing function of the queue-length [12], [14], [13].) Then, the resulting π_t, a solution to $\pi = \pi P_t$, is also slowly varying over time t such that the actual distribution μ_t is able to get closer to π_t (in the sense of TV distance) before π_t moves away to another, thus effectively simulating the separation of timescales. As will be shown in Section IV-C, under our algorithm with order parameter T, the speed of convergence of μ_t under static fugacity (i.e., static P) is roughly T times slower than that of the conventional Glauber dynamics. This implies that the speed of actual distribution μ_t under dynamic fugacity as a slowly varying function of queue-length, will also be reduced by a factor of T. Since T is finite, we expect that μ_t is still able to catch up the slowly varying target π_t in time, and by following similar steps in [12], [14], [13] our algorithm can also achieve the throughput optimality. Our result below shows that this is indeed the case.

Proposition 5. *Let $\epsilon > 0$ be arbitrarily given. For any arrival rate $\boldsymbol{\eta} \in (1-\epsilon)\mathbb{C}$, set the dynamic link weight*[2]

$$W_v(t) = \max\left\{h(Q_v(t)), \frac{\epsilon}{2|\mathcal{N}|}h(Q_{max}(t))\right\}, \qquad (10)$$

where $h(\cdot) = \log\log(\cdot + e)$. Then, our algorithm with any finite order parameter T satisfies (3), and is thus throughput optimal.

C. Impact on transient behavior

So far, we have focused on the study of correlations and throughput-delay performance in the steady-state. As seen earlier, our algorithm yields smaller delay once the system has converged to its stationary regime, and larger T in our algorithm leads to better queueing performance in the steady-state. While such a convergence itself in a finite-sized system is always guaranteed, it is unclear how our algorithm affects such transient behavior and the speed of convergence to its stationary regime. To capture such transient behavior, we utilize a notion of *mixing time* of a Markov chain P with its stationary distribution π, defined as follows [35], [34], [42]

$$t_{\text{mix}}(\epsilon) \triangleq \min\{t : \max_{x\in\Omega} \|P^{(n)}(x,\cdot) - \pi\|_{\text{TV}} \le \epsilon, \ \ \forall n \ge t\},$$

where $P^{(n)}(x,\cdot)$ is n-step transition probability distribution starting from x. For general, non-Markov processes with the same stationary distribution π on the same state space, the mixing time can be similarly defined by considering $\|\mu_t - \pi\|_{\text{TV}}$, where μ_t is the distribution at time t and maximizing over all possible initial distribution μ_0. Clearly, our delayed CSMA algorithm with order parameter T yields $\mu_t = \mu_{t-T}P$ when the transition kernel is time-homogeneous. Iteratively applying such relation gives $\mu_{nT} = \mu_0 P^{(n)}$, implying that in terms of the mixing time we need roughly T times larger number of state transitions to achieve the same degree of accuracy in distributional error, compared to the conventional CSMA-based algorithm. In other words, in our high-order Markov chain based approach, while the stationary distribution of the schedule itself remains untouched and it provides better queueing performance in the steady state, it may take roughly T times longer to reach the steady state.[3] This tradeoff between a bit slower convergence but to a 'better' stationary regime, also implies that the performance during the transient phase can be worse than that of the conventional CSMA algorithm (albeit our algorithm will eventually pay off in the end), and necessitates a careful choice of T depending upon how long the system is meant to be running and measured.

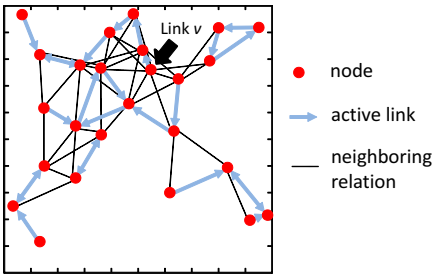


Fig. 5. An instance of network configuration with 25 nodes

V. Simulation Results

In this section, we present simulation results for the delayed CSMA algorithm. We consider a network scenario by a random geometric graph (RGG) model with 25 nodes uniformly and independently positioned over the $1000 \times 1000m^2$ area. The transmission range of each node is set to be $250m$, where two nodes can communicate with each other if they are in the range of transmission. We have each node select one of its neighboring node uniformly at random, and create a communication link. If the receiver node of a link is within a range of transmitter node of another link, we consider that the two links form an edge in the conflict graph. For the decision schedule, we choose the access probabilities $a_i = 0.25$ for links $i \in \mathcal{N}$. In this scenario, we collect simulation results for the link v shown in Figure 5. In obtaining simulation data, we have taken average of the data from 10000 time repeated simulations, and unless otherwise noted, for each of the runs we discard the data from the first half of the simulation time, to obtain the results in the steady-state.

First, to understand the benefits of having more drastic changes in the zero-one service process $\sigma_v(t)$ under our delayed CSMA algorithm, we look at the distribution of its 'off' duration U, the duration from an active slot to the next active slot. We measured its mean $\mathbb{E}\{U\}$ and the coefficient of variation (CoV) $\sqrt{\text{Var}\{U\}}/\mathbb{E}\{U\}$ as we increase the order parameter T, where we set $\lambda_i = 1$ for all $i \in \mathcal{N}$. Figure 6 shows that the first-order statistics of the link state doesn't change with T, while its variability decreases for larger T, implying that our algorithm with larger T effectively removes link starvation phenomenon.

[2] $Q_{\max}(t) = \max_{i\in\mathcal{N}} Q_i(t)$. In [12] it is argued that $Q_{\max}(t)$ can be easily estimated a gossip-like message passing mechanism. For simplicity, we assume $Q_{\max}(t)$ is known throughout, and do not discuss this issue here.

[3] This also suggests that the mixing time based delay analysis may give looser bound on the average queue length *in the steady-state*. See [20] for similar accounts on the usage of mixing time for delay analysis.

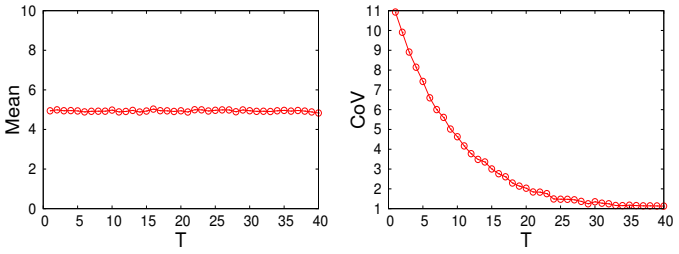
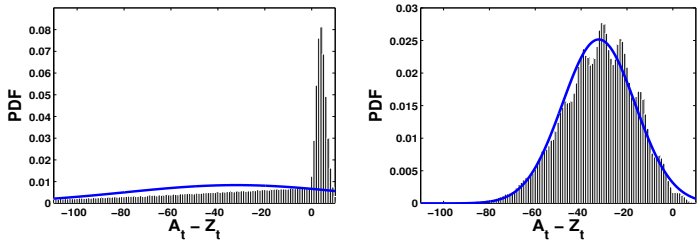


Fig. 6. Mean and CoV of 'off' duration for $\sigma_v(t)$



(a) T=1, t=217 (b) T=25, t=217

Fig. 7. Histogram of $\mathbf{A}_t - \mathbf{S}_t$ and its approximation by a Gaussian process. The lines are Gaussian distribution drawn from measured sample mean and variance.

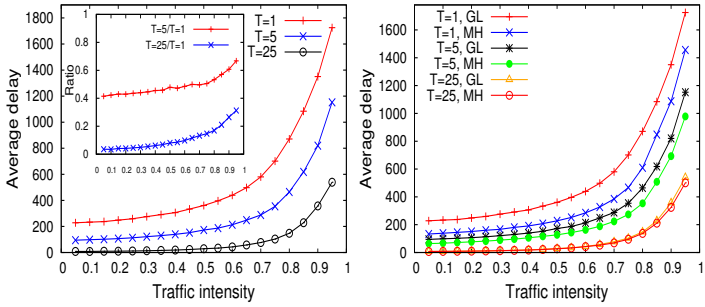
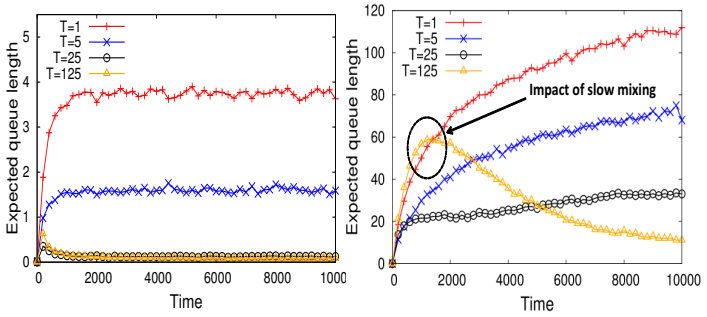


Fig. 8. Delay performance of the delayed CSMA with various T under dynamic fugacity



(a) Traffic intensity=0.1 (b) Traffic intensity=0.9

Fig. 9. Impact of T on a queue evolution.

Before investigating the delay performance of our proposed algorithm, we first validate our approximation. As before, under the static fugacity setup, we first collect the cumulative net-input $\mathbf{A}_t - \mathbf{S}_t$ up to $t = 217$ from a stationary start $t = 0$ where the arrival rates are all set to be $\eta_v = 0.1$. As seen in Figure 7(a), the net-input process in the standard CSMA case ($T = 1$) is far from being Gaussian due to the strong correlation structure in $\sigma_v(t)$. As we increase the order parameter to $T = 25$, it becomes very close to Gaussian as shown in Figure 7(b). The same conclusion also applies to the modified net-input $\mathbf{Z}(t) = \mathbf{A}_t - \mathbf{S}_t + Ct$.

In measuring delay performance, we consider dynamic fugacity with the weight function set to be $\log\log(Q_i(t) + e)$ for all $i \in \mathcal{N}$. We have measured the delay of each packet in the queue until gets served, under different traffic intensity. We adjusted arrival rate of each link by gradually increasing the rate proportional to its potential link capacity, which is calculated by summing over all possible maximal independent sets with equal weight. As the traffic intensity increases to 1, the rate approaches to maximum throughput. The left-hand side of Figure 8 shows the delay improvement over the conventional CSMA algorithm, where the inset figure displays the ratio of the delay under our algorithm with chosen T to that of standard CSMA. We note that the performance improvement is quite remarkable. For instance, with $T = 5$, the delay reduces by half, and with $T = 25$, it reduces by a factor of 20 compared to the conventional CSMA algorithm. We observed similar trends in the improvement under different weigh functions such as $\log(Q_i(t) + 1)$ or $Q_i(t)$.

Recall that our approach is to add 'order of T' into the original Glauber dynamics to keep the same marginal distribution while shaping the correlation structure to our advantage. We note that this approach will hold for any other standard algorithm modeled by a reversible Markov chain on the same feasible state space achieving the same stationary distribution, not necessarily the Glauber dynamics considered so far. For example, it was shown in [19] that Metropolis-Hastings (MH) based algorithm (still reversible Markov chain on Ω) outperforms the conventional Glauber dynamics. The right-hand side of Figure 8 shows that indeed MH algorithm gives better delay performance than the standard Glauber dynamics (GL), but still, the amount of delay reduction is significant for larger T. Interestingly, the performance gap between the standard Glauber and the MH algorithm (all with $T = 1$) seems to narrow down for larger T, implying that the effect of correlation reductions via high-order Markov chain dominates over the choice of base reversible Markov chains.

We now look at the impact of transient behavior induced by our algorithm with different T. We have measured the queue-length from after simulation starts. In order to measure the queue-length during the initial phase, we collect queue-length samples and obtain its time average for different time indexes. Figure 9 shows the measured queue-length under different T. As discussed in section IV-C, we might expect larger backlog at initial phase when large T is used (due to slower mixing time). However, in practice, its impact is not significant unless indefinitely large T is used. For example, as can be seen from Figure 9(a), its impact is almost negligible when traffic intensity is low. Even under high traffic intensity (Figure 9(b)) and with large T, e.g., $T = 125$, the accumulation of backlog at transient phase due to slower mixing is still comparable to that of the original case ($T = 1$), and the queue eventually converges to smaller size in the steady state. Thus clearly, the benefit in long-term behavior outweighs such a drawback.

VI. Conclusion

In this paper, we have proposed the delayed CSMA algorithm based on a high-order Markov chain on the same state space of feasible schedules achieving the same stationary distribution, but with different and reshaped correlation structure. Our algorithm is extremely simple to implement without requiring any additional message passing or overhead, with the exception of a single parameter T across the network once and for all. We have proved that our algorithm is throughput-optimal, yet provides far better delay performance by emulating the effect of superposition of independent traffic streams in the queue, out of a single input and still single service process, via zero-padding and lag-shifting properties of the resulting service process of the queue under our algorithm.

Another interesting viewpoint arising from our investigation is that the conventional approach via mixing time for delay performance must be used with great care, since in our setting we achieve better queueing performance in the steady-state by trading much less correlations for a bit slower mixing speed. While faster mixing and less correlations typically come in pair under the traditional Markov chain based approach, we note that our high-order Markov chain (or simply non-Markov on the same state space) can now separate these two, thus enabling us to trade one for the other, toward better understanding and design of distributed schedulers running over different timescales of interest.

References

[1] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling for maximum throughput in multihop radio networks," *IEEE Trans. on Automatic Control*, vol. 37, no. 12, pp. 1936–1949, Dec. 1992.

[2] C. Joo, X. Lin, and N. B. Shroff, "Understanding the capacity region of the greedy maximal scheduling algorithm in multi-hop wireless networks," in *IEEE INFOCOM*, April 2008.

[3] X. Wu, R. Srikant, and J. R. Perkins, "Scheduling efficiency of distributed greedy scheduling algorithms in wireless networks," in *IEEE INFOCOM*, May 2006.

[4] A. Dimakis and J. Walrand, "Sufficient conditions for stability of longest queue first scheduling: Second order properties using fluid limits," *Adv. Appl. Probab.*, vol. 38, no. 2, pp. 505–521, 2006.

[5] G. Zussman, A. Brzezinski, and E. Modiano, "Multihop local pooling for distributed throughput maximization in wireless networks," in *IEEE INFOCOM*, April 2008.

[6] M. Leconte, J. Ni, and R. Srikant, "Improved bounds on the throughput efficiency of greedy maximal scheduling in wireless networks," in *ACM MobiHoc*, May 2009.

[7] A. Brzezinski, G. Zussman, and E. Modiano, "Enabling distributed throughput maximization in wireless mesh networks a partitioning approach," in *ACM MobiCom*, September 2006.

[8] E. Modiano, D. Shah, and G. Zussman, "Maximizing throughput in wireless networks via gossiping," in *ACM SIGMETRICS/Performance*, June 2006.

[9] S. Sanghavi, L. Bui, and R. Srikant, "Distributed link scheduling with constant overhead," in *ACM Sigmetrics*, June 2007.

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

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

[12] S. Rajagopalan, D. Shah, and J. Shin, "Network adiabatic theorem: An efficient randomized protocol for contention resolution," in *ACM SIGMETRICS/Performance*, Seattle, WA, June 2009.

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

[14] J. Ghaderi and R. Srikant, "On the design of efficient csma algorithms for wireless networks," arXiv report http://arxiv.org/abs/1003.1364, 2010.

[15] D. Shah and J. Shin, "Delay optimal queue-based CSMA," in *ACM SIGMETRICS*, Columbia University, NY, June 2010.

[16] M. Lotfinezhad and P. Marbach, "Throughput-optimal random access with order-optimal delay," in *IEEE INFOCOM*, April 2011.

[17] 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 *IEEE ISIT*, July 2012.

[18] P.-K. Huang and X. Lin, "Improving the delay performance of CSMA algorithms: a virtual multi-channel approach," in *IEEE INFOCOM*, April 2013.

[19] C.-H. Lee, D. Y. Eun, S.-Y. Yun, and Y. Yi, "From glauber dynamics to metropolis algorithm: Smaller delay in optimal csma," in *IEEE ISIT*, July 2012.

[20] V. Subramanian and M. Alanyali, "Delay performance of CSMA in networks with bounded degree conflict graphs," in *IEEE ISIT*, July 2011.

[21] 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 Trans. on Networking*, vol. 20, no. 3, pp. 825–836, June 2012.

[22] J. Ghaderi and R. Srikant, "Effect of access probabilities on the delay performance of q-csma algorithms," in *IEEE INFOCOM*, April 2012.

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

[24] S. P. Meyn and R. L. Tweedie, "Criteria for stability of Markovian processes I: Discrete time chains," *Advances in Applied Probability*, vol. 24, pp. 542–574, 1992.

[25] A. Ganesh, N. O'Connell, and D. Wischik, *Big Queues*. Springer, 2004.

[26] F. Bacelli and P. Brémaud, *Elements of Queueing Theory: Palm Martingale Calculus and Stochastic Recurrences*. Springer-Verlag, 2003.

[27] R. G. Addie and M. Zukerman, "An Approximation for Performance Evaluation of Stationary Single Server Queues," *IEEE Transactions on Communications*, vol. 42, no. 12, pp. 3150–3160, Dec. 1994.

[28] N. G. Duffield, "Exponential bounds for queues with Markovian arrivals," *Queueing Systems*, vol. 17, pp. 413–430, 1994.

[29] N. G. Duffield and N. O'Connell, "Large deviations and overflow probabilities for the general single server queue, with application," *Proc. Cambridge Philos. Soc.*, vol. 118, pp. 363–374, 1995.

[30] F. Kelly, "Notes on effective bandwidths," *Stochastic networks: Theory and Applications*, Oxford University Press, 1996.

[31] J. Kwak, C.-H. Lee, and D. Y. Eun, "A high-order markov chain based scheduling algorithm for low delay in csma networks," Technical Report, http://www4.ncsu.edu/~dyeun/pub/techrep-delayed-csma13.pdf.

[32] C. J. Geyer, "Practical Markov Chain Monte Carlo," *Statistical Science*, vol. 7, no. 4, pp. 437–483, Nov. 1992.

[33] D. Aldous and J. Fill, *Reversible Markov Chains and Random Walks on Graphs*. monograph in preparation.

[34] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov chains and mixing times*. American Mathematical Society, 2009.

[35] P. Brémaud, *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*. Springer-Verlag, 1999.

[36] R. M. Loynes, "The stability of a queue with non-independent inter-arrivals and service times," *Cambridge Phil.*, vol. 58, pp. 497–520, 1962.

[37] D. D. Botvich and N. Duffield, "Large deviations, the shape of the loss curve, and economies of scale in large multiplexers," *Queueing Systems*, vol. 20, pp. 293–320, 1995.

[38] A. Hordijk, Z. Liu, and D. Towsley, "Smoothing effect of the superposition of homogeneous sources in tandem networks," *Journal of Applied Probability*, vol. 37, no. 3, pp. 900–913, 2000.

[39] K. Kumaran, M. Mandjes, and A. Stolyar, "Convexity properties of loss and overflow functions," *Operations Research Letters*, vol. 31, no. 2, pp. 95 – 100, 2003.

[40] J. Choe and N. B. Shroff, "Use of supremum distribution of gaussian processes in queueing analysis with long-range dependence and self-similarity," in *Stochastic Models*, vol. 16, no. 2, Feb. 2000.

[41] D. Y. Eun and N. B. Shroff, "A measurement-analytic approach for QoS estimation in a network based on the dominant time scale," *IEEE/ACM Trans. on Networking*, vol. 11, no. 2, pp. 222–235, Apr. 2003.

[42] D. W. Stroock, *An Introduction to Markov Processes*. Springer, 2005.