

# An Antithetic Coupling Approach to Multi-Chain based CSMA Scheduling Algorithms

Jaewook Kwak

Do Young Eun

**Abstract**—In recent years, a suite of Glauber dynamics-based CSMA algorithms have attracted great attention due to their simple, distributed implementations with guaranteed throughput-optimality. However, these algorithms often suffer from poor delay performance and the starvation problem. Among several attempts to improve the delay performance, a remarkable improvement has recently been made in a class of CSMA algorithms that utilize multiple instances of the algorithm (or Markov chains). In this paper, we develop a new approach via an antithetic coupling (AC) method, which can further improve the delay performance of those that virtually emulate multiple chains. The key enabler of utilizing AC method lies in our skilful choice of manipulating the driving sequences of random variables that govern the evolution of schedule instances, in such a way that those multiple instances of chains become *negatively correlated* as oppose to having them run independently. This contributes faster change of the link state, rendering it more like a periodic process and thus leading to better queueing performance. We rigorously establish an ordering relationship for the effective bandwidth of each net-input process to the queue, between our proposed algorithm (AC-CSMA) and other state-of-the-art existing algorithms in the literature, under a mild set of assumptions. The proposed algorithm involves very simple modification onto existing CSMA-based algorithms, and can be implemented in a fully distributed manner without any additional message overhead. Our extensive simulation results also confirm that AC-CSMA always delivers better queueing performance over a variety of network scenarios.

## I. INTRODUCTION

In wireless networks, packet scheduling and queue management play an important role in achieving efficient utilization of wireless resources and providing satisfactory quality-of-service. Designing such good *scheduling algorithms*, however, is a challenging problem, as users in a wireless network may experience a complicated contention relationship among the participants. In an early work by Tassiulas and Ephremides [1], a scheduling algorithm known as *Maximum Weight Scheduling* (MWS) was proposed to achieve the maximum capacity, but it requires to solve a complicated combinatorial problem based on global information. Subsequent works have attempted to address practical issues, but they often suffer loss of capacity or require a huge amount of message overhead [2].

Recently, a certain class of CSMA-based scheduling algorithms have gained great attention as a new solution to the design of scheduling algorithm. These algorithms are based on a so-called Glauber dynamics, which enables to find the

schedules that are close to the max-weight schedules. The main appealing feature of the CSMA-based algorithms is that they effectively achieve the throughput optimality while imposing only little or no message overhead. For example, Jiang and Walrand [3] characterized the achievable throughput of the CSMA algorithm as a function of locally adjustable control parameters and developed an algorithm that adaptively chooses these parameters. They showed that the algorithm achieves the throughput optimality under a time scale separation assumption, i.e., the schedule dynamics converges to its steady state quicker than the time-scale of parameter adaptation. In a similar spirit, several approaches have been devised for adapting the parameter based on queue-length information, and have been shown to guarantee the stability or the throughput optimality [4], [5], [6].

Despite the advantage of simplicity in implementation, the CSMA algorithms often exhibit large delay. The major reason for the poor performance is due to the fundamental constraint imposed from schedule dynamics. Specifically, the current schedules by the CSMA algorithms are constrained to change to a limited set of feasible schedules next. This leads to the *starvation* phenomenon for the link service processes, i.e., links tend to be unable to obtain transmission time slots for a long time. This issue has triggered a strong interest in improving the delay performance [7], [8], [9], [10].

A notable approach to deal with the poor delay performance of the CSMA algorithms is to utilize multiple instances of such schedulers. For example, in [11], the authors consider the CSMA algorithms with multiple channels and quantify the effect of the number of channels on the starvation problem. Huang and Lin [12] propose a virtual multiple channel scheme to maximize the aggregate of utilities of links, while achieving asymptotically bounded head-of-line waiting time. In [13], Kwak et al. propose a so-called *delayed CSMA* algorithm that can effectively emulate such an effect by utilizing past history of link schedules, and deliver sizable performance improvement in terms of reducing average queue sizes. Later, it has been shown that this algorithm can be harnessed to achieve constant-bounded time-averaged queueing delay [14].

In this paper, we focus on the delayed CSMA algorithm [13] and propose a new approach that can further improve its delay performance by incorporating the antithetic coupling (AC).<sup>1</sup> The AC method has been known as an efficient way to improve the accuracy in multiple MCMC simulations [15], [16], [17], [18]. The underlying idea is, for some sample paths generated,

The authors are with Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27695-7911, Email: {jkwak, dyeyun}@ncsu.edu.

This work was supported in part by National Science Foundation under Grant No. CNS-1217341 and CNS-1423151.

<sup>1</sup>It is also referred to as antithetic variates method in the statistics literature.

to take advantage of its antithetic path to reduce variance. In a nutshell, the delayed CSMA algorithm can be viewed as if it runs multiple instances of Markov chains, and thus constitutes a suitable setup on which we can apply AC approach so as to improve the queueing performance. Our rationale behind applying AC idea is to *negatively correlate* such multiple chains, going beyond having them run independently. By doing so, the resulting service process of each individual link tends to be closer to a *periodic* process, leading to better delay performance overall. However, it is uncertain how precisely we can utilize AC idea *at no additional cost* and whether it really leads to provable better queueing performance. This question arises because the AC method may even worsen the performance when wrongly applied [17], and thus it has to be used with great care.

To set the stage, we first look into the performance implications expressed by the correlation structure of the standard CSMA. This gives insights on why CSMA algorithms suffer from poor delay performance. Specifically, we formally prove that all the eigenvalues of CSMA algorithms (Markov chains) are non-negative, implying that the service process of any link by the CSMA algorithm is positively correlated *over all time*. As a goal of this paper, we then aim at showing that the proposed algorithm provides guaranteed performance improvement. To accomplish this, we resort to the *effective bandwidth* of the net-input process, which quantifies the degree of queue occupancy by incorporating the dynamics of arrival and service processes altogether. (More precise statements will be given in Section V.) We prove that utilizing the AC method in the way we propose here indeed achieves smaller effective bandwidth, which in turn leads to better queueing performance and smaller delay. The biggest difficulty in our analysis lies in the fact that the configurational state space of the involved CSMA algorithm (set of feasible schedules or independent sets of the conflict graph) defies the conventional notion of monotonicity – a necessary ingredient to create antithetically coupled sample paths. Nonetheless, we are able to prove the desired performance ordering under a reasonable set of assumptions on the graph topologies. The assumptions are purely for technical purpose, and we believe that they are not necessary in practical settings. This argument is further supported by our extensive simulation results on a wide range of network scenarios.

The rest of this paper is organized as follows. In Section 2, we present our network model and preliminaries on the CSMA scheduling, as well as the delayed CSMA algorithm. In Section 3, we provide our theoretical findings that were unknown to some of previous works. In particular, our results reveal that strong correlations indeed persist in the link state process. In Section 4, we first introduce the concept of antithetic coupling and investigate several methods that can practically implement the idea. We then present our main idea with a motivating example and provide detailed procedure of our proposed algorithm. In Section 5, we provide analysis on showing that the proposed algorithm achieves smaller effective bandwidth, which in turn leads to better queueing performance. Section

6 presents our extensive simulation results under various network scenarios, and Section 7 concludes the paper.

## II. PRELIMINARIES

### A. System Model

Consider a wireless network with a conflict graph  $\mathcal{G} = (\mathcal{N}, \mathcal{E})$  where  $\mathcal{N}$  is the set of links (or nodes in the conflict graph<sup>2</sup>) and  $\mathcal{E}$  is the set of edges. An edge  $(i, j) \in \mathcal{E}$  exists between two links  $i$  and  $j$  if they cannot be active at the same time due to their mutual interference. Define by  $\sigma = (\sigma_v)_{v \in \mathcal{N}} \in \{0, 1\}^{|\mathcal{N}|}$  that represents the set of link states. A link  $v$  is active if it is included in the schedule, i.e.,  $\sigma_v = 1$ , and is inactive if otherwise. A schedule is called to be feasible if the links in the schedule can be active at the same time slot according to the conflict graph relationship  $\mathcal{G}$ . Evidently, a feasible schedule  $\sigma$  should satisfy the independent set constraint i.e.,  $\sigma_v + \sigma_w \leq 1$  for all  $(v, w) \in \mathcal{E}$ . We denote by  $\Omega$  the set of all feasible schedules. We assume the network runs in a time-slotted manner, and therefore we denote by  $\sigma(t)$  the schedule instance at time  $t$  for  $t = 0, 1, \dots$

Each link is associated with a queue fed by some exogenous traffic arrivals and serviced when the link is active. We consider that  $A_v(t)$  amount of packets arrive to the queue of link  $v$  at each time slot  $t$  in an i.i.d. manner according to some distribution provided that  $\mathbb{E}\{A_v(t)\} = \nu_v$  and  $\text{Var}\{A_v(t)\} = \nu_v^* < \infty$ . Let  $\nu = (\nu_v)_{v \in \mathcal{N}}$  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:

$$Q_v(t) = \max\{Q_v(t-1) + A_v(t) - \sigma_v(t), 0\}, \quad t \geq 1. \quad (1)$$

The capacity region of the network is the set of all arrival rates  $\nu$  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.,

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

and, an algorithm is called *throughput-optimal* if it can maintain all the queues in the network finite for any arrival rates within the capacity region.

### B. Glauber Dynamics and CSMA Scheduling

While the MWS algorithm has been known to achieve the throughput-optimality [1], it is not a practical solution because it requires to solve a complicated combinatorial problem every time slot. Recently, several CSMA algorithms have emerged that offer the capability of achieving the full capacity region and thus match the optimal throughput performance of the MWS. Central to these CSMA algorithms is the so-called *Glauber dynamics*, which is a method of sampling independent sets with a desired probability distribution. The traditional Glauber dynamics works as follows. At each time slot  $t$ , links select a set of links  $\mathbf{m}(t)$  that is interference free in

<sup>2</sup>We will use the term ‘node’ and ‘link’ interchangeably.

a distributed manner. For those selected links  $v \in \mathbf{m}(t)$ , if the link senses the transmission of any conflicting link, then it keeps silent. If none of its conflicting links is transmitting, then it transmits data with probability  $\frac{\lambda_v}{1+\lambda_v}$ .

In this setup, the schedule  $\sigma(t)$  forms a Markov chain which is irreducible, aperiodic, and reversible over  $\Omega$ , and achieves the stationary distribution given by  $\pi(\sigma) = \frac{1}{Z} \prod_{i \in \mathcal{N}} \lambda_i^{\sigma_i}$  where  $Z = \sum_{\sigma \in \Omega} \prod_{i \in \mathcal{N}} \lambda_i^{\sigma_i}$  is a normalizing constant. A link  $v \in \mathcal{N}$  then receives an average service rate  $s_v = \lim_{s \rightarrow \infty} \frac{1}{s} \sum_{t=1}^s \sigma_v(t) = \sum_{\sigma: \sigma_v=1} \pi(\sigma)$ . The fundamental idea for establishing the throughput-optimality comes from the fact that there exists  $\{\lambda_v\}_{v \in \mathcal{N}}$  such that the average service rate is strictly larger than the average arrival rate, i.e.,  $\nu_v < s_v$ ,  $\forall v \in \mathcal{N}$ . There have been several approaches that adaptively find appropriate values for the  $\lambda$ 's using queue information [4], [5] or experienced arrival and service rate [19].

### C. Delayed CSMA

The delayed CSMA algorithm runs with a parameter  $T \geq 1$  which captures the number of chains used for deciding link activities. If  $T = 1$ , the algorithm becomes the conventional CSMA algorithm, and hence Algorithm 1 can be understood as a generalized version of it.

---

#### Algorithm 1 Delayed CSMA [13]

---

```

1: Initialize: for all links  $v \in \mathcal{N}$ ,  $\sigma_v(t) = 0$ ,  $t = 0, 1, \dots, T-1$ .
2: At each time  $t \geq T$ : links find a decision schedule,  $\mathbf{m}(t)$  through
   a randomized procedure, and
3: for all links  $v \in \mathbf{m}(t)$  do
4:   if  $\sum_{w \in N(v)} \sigma_w(t-T) = 0$  then
5:      $\sigma_v(t) = 1$  with probability  $\frac{\lambda_v}{1+\lambda_v}$ 
6:      $\sigma_v(t) = 0$  with probability  $\frac{1}{1+\lambda_v}$ 
7:   else  $\sigma_v(t) = 0$  end if
8: end for
9: for all links  $w \notin \mathbf{m}(t)$  do
10:   $\sigma_w(t) = \sigma_w(t-T)$ 
11: end for

```

---

In the conventional CSMA algorithm, i.e., Algorithm 1 with  $T = 1$ , the schedule may easily get trapped in a particular subset of feasible schedules for a long period of time. This phenomenon, a.k.a. temporal starvation problem, is due to the fundamental constraint imposed by the Markov chain that limits state transition from one schedule to another. The delayed CSMA algorithm (with  $T \geq 2$ ) enables to effectively resolve this problem by allowing more drastic changes of the schedules over time.

There is a certain trade-off in choosing the parameter  $T$  in the delay performance. As far as the steady state performance is of a concern, larger  $T$  is always preferable. A recent work has shown that this algorithm can be delay-optimal in that there exists some  $T$  such that the average delay can be bounded by a constant [14]. However, larger  $T$  entails longer time for scheduling instances to reach steady state, resulting in degrading the performance in the transient phase [13]. In this paper, we propose a new algorithm, called AC-CSMA, that can provide better delay performance over the delayed CSMA with any given  $T \geq 2$  in use. Our idea is based on the concept of

antithetic coupling that has been widely used in the statistics literature. In Section IV, we will briefly discuss its concept and how to utilize it in the CSMA setup.

### III. REVISITING CORRELATION STRUCTURE OF THE STANDARD CSMA ALGORITHM

According to the queueing theory literature in general, the variability of arrival and service processes significantly affects its queueing performance. In particular, it has been reported through a number of works in the literature that the positive correlation of the arrival and/or service process has an adverse impact on the queueing delay [20], [21], [22]. From this viewpoint, understanding the correlation structure is pivotal in analyzing the performance of a queueing system. Thus, we here investigate the correlations of the link service process modulated by the standard CSMA algorithm (or the delayed CSMA with  $T = 1$  in Algorithm 1).

Let  $\sigma(t) \in \Omega$  be a Markov chain that represents a feasible schedule by the CSMA algorithm at time slot  $t$ . Since we are interested in the long-term behavior of queueing performance, we assume that the Markov chain  $\sigma(t)$  is in its stationary regime, i.e.,  $\mathbb{P}\{\sigma(t) = \mathbf{x}\} = \pi(\mathbf{x})$  for all  $t \geq 0$ . For any given function  $f : \Omega \rightarrow \mathbb{R}$ , we define the correlation coefficient of lag  $k$  by  $\psi(f, k) \triangleq \text{Corr}\{f(\sigma(t)), f(\sigma(t+k))\}$ , where  $\text{Corr}\{X, Y\} = \frac{\text{Cov}\{X, Y\}}{\sqrt{\text{Var}\{X\}}\sqrt{\text{Var}\{Y\}}}$ . Let  $B_v$  be the set of all states for which a link  $v$  is active, i.e.,  $B_v \triangleq \{\mathbf{x} \in \Omega : \mathbf{x}_v = 1\} \subset \Omega$ . We can then write the correlation at lag  $k$  for the service process of a particular link  $v$  by  $\psi(\mathbf{1}_v, k)$ , where  $\mathbf{1}_v(\mathbf{x}) \triangleq \mathbf{1}\{\mathbf{x} \in B_v\}$  for  $\mathbf{x} \in \Omega$ .

In [23], it has been shown that for any finite state reversible Markov chain, the correlations at even-lags, i.e.,  $\psi(f, k)$ , for  $k = 2, 4, \dots$  are all positive. For the CSMA algorithm, the authors in [13] have shown that the degree of correlation for the service process is quite high for a wide range of time scales by quantifying the correlations for lag-1 and even-lags. A natural question is whether the same is true for all odd-lags other than lag-1.

In the statistics literature, we found in [24] that for a class of Gibbs sampler,<sup>3</sup> the positive correlations hold for all lags and for any bounded function  $f : \Omega \rightarrow \mathbb{R}$  and over a finite state space  $\Omega$ . The authors in [24] focused only on a unitary random scan scheme, i.e., updating a single entity at random, which translates into choosing only a single link as a decision schedule at every time slot. In CSMA, however, the network decides multiple links to update in a single time slot [19], [26], [27], [13], which is not covered by their analysis. We here verify that similar results hold for the multi-site update cases. We consider a class of multi-site update schemes that select an independent set of nodes at every time i.i.d. with some probability. To be more specific, the network selects a decision schedule  $\mathbf{m}(t) \in \Phi$  at every time  $t$  i.i.d. with probability  $\alpha(\mathbf{m})$  where  $\sum_{\mathbf{m} \in \Phi} \alpha(\mathbf{m}) = 1$ . For example, if each link  $i$  can attempt the channel with access probability  $a_i$ , the link  $i$  is then selected if it is interference free among its

<sup>3</sup>Glauber dynamics is a special instance of Gibbs sampler [25].

neighbors. The resulting decision schedule clearly forms an independent set.

**Proposition 1.** *For any function  $f : \Omega \rightarrow \mathbb{R}$ , it holds  $\psi(f, 1) \geq 0$ .*

*Proof.* See Appendix.  $\square$

[13] shows the positive correlation of lag-1 for the service process, i.e.,  $\psi(f, 1) \geq 0$  for  $f = \mathbf{1}_v$  only, while Proposition 1 holds for any arbitrary function  $f$ . This extension is in fact strong enough to characterize the entire spectrum of the CSMA algorithm as shown next.

**Proposition 2.** *Let  $\rho_j$ ,  $j = 1, \dots, |\Omega|$  be the left eigenvalues of the transition probability matrix of the standard CSMA algorithm. Then  $\rho_j \geq 0$  for all  $j = 1, \dots, |\Omega|$ .*

*Proof.* Let  $(\rho_j, \mathbf{v}_j)$ ,  $j = 1, 2, \dots, |\Omega|$ , be the  $j^{\text{th}}$  eigenvalue and eigenvector pair of the transition probability matrix. For any two functions  $f, g : \Omega \rightarrow \mathbb{R}$  (or  $|\Omega|$ -dimensional vectors), define  $\langle f, g \rangle_\pi = \sum_{i=1}^{|\Omega|} f(i)g(i)\pi(i)$ . Since the chain  $\sigma(t)$  is reversible, we have  $\langle \mathbf{v}_i, \mathbf{v}_j \rangle_\pi = 0$  for  $i \neq j$ , i.e., they are orthogonal [28]. Define a random variable  $\Lambda$  such that

$$\mathbb{P}\{\Lambda = \rho_j\} = |\langle f, \mathbf{v}_j \rangle_\pi|^2 / C, \quad j = 2, \dots, |\Omega|$$

where  $C = \sum_{j=2}^{|\Omega|} |\langle f, \mathbf{v}_j \rangle_\pi|^2 > 0$  is a normalizing constant. Now, we utilize the spectral characterization of the correlation function as shown in [29]. Specifically, for any ergodic reversible chain, Lemma 1 in [29] states that  $\psi(f, k) = \mathbb{E}\{\Lambda^k\}$ . Thus, we have  $\psi(f, 1) = \mathbb{E}\{\Lambda\}$  and our Proposition 1 guarantees that this is non-negative for any choice of  $f$ . Set  $f = \mathbf{v}_j$ . The result then follows by noting that  $\{\mathbf{v}_i\}$  are orthogonal and thus  $\mathbb{P}\{\Lambda = \rho_j\} = 1$ , i.e.,  $\mathbb{E}\{\Lambda\} = \rho_j \geq 0$ .  $\square$

By using the above two Propositions, we can now show that the correlations at all lags are positive.

**Proposition 3.** *For any  $f : \Omega \rightarrow \mathbb{R}$ ,  $\psi(f, k) \geq 0$ ,  $\forall k \geq 1$ .*

*Proof.* Proposition 2 asserts that the random variable  $\Lambda$  takes on non-negative values, i.e.,  $\Lambda \geq 0$ . Thus,

$$\psi(f, k) = \mathbb{E}\{\Lambda^k\} \geq \mathbb{E}\{\Lambda\}^k = \psi(f, 1)^k \geq 0,$$

where the first inequality is from the Jensen's inequality and the second inequality is from Proposition 1.  $\square$

Our findings here have several implications on some of previous works. In [8], the authors study the impact of decision schedules on the delay performance where their analysis is based on the assumption that the second largest eigenvalue in modulus (SLEM),  $\max\{\rho_2, |\rho_{|\Omega|}|\}$ , is equivalent to  $\rho_2$ . Our Proposition 2 verifies that the SLEM is indeed equal to  $\rho_2$  without any assumption. Also, in [13], the improvement on delay performance of delayed CSMA was demonstrated by the smaller variance of the cumulative service process, based on the assumption that correlations of a link service process are all non-negative for all lags. Again, our Proposition 3 indicates that such assumption is actually not necessary.

#### IV. ANTITHETIC COUPLING APPROACH TO DELAYED-CSMA ALGORITHM

In this section, we propose our main algorithm that is based on a novel concept of antithetic coupling. We start this section by briefly introducing its concept, and then present our idea on how to utilize it in CSMA scheduling.

##### A. Antithetic coupling

The antithetic coupling method is a variance reduction technique used in parallel Monte Carlo simulations [15], [16]. The key principle is, for some sample paths generated, to take advantage of its antithetic path to reduce variance. To illustrate, consider two Markov chains,  $X_t$  and  $Y_t$ , each with the same state space  $\Omega$  and transition probability  $\mathbf{P}$ . Suppose that each chain has produced  $t$  samples after sufficient amount of burn-in time to remove dependency from its initial state. A natural estimator that combines the two set of  $t$  samples would be

$$\hat{\theta}(f, t) = (S_t^X + S_t^Y) / 2t.$$

where  $S_t^X = \sum_{i=0}^{t-1} f(X_i)$  and  $S_t^Y = \sum_{i=0}^{t-1} f(Y_i)$ . In case where the two sample paths are independent, the variance is

$$\begin{aligned} \text{Var}(\hat{\theta}(f, t)) &= [\text{Var}(S_t^X) + \text{Var}(S_t^Y) + 2\text{Cov}(S_t^X, S_t^Y)] / 4t^2 \\ &= [\text{Var}(S_t^X) + \text{Var}(S_t^Y)] / 4t^2 = \text{Var}(S_t^X) / 2t^2. \end{aligned}$$

The antithetic method is to generate the second sample path  $Y_0, \dots, Y_{t-1}$  in a way that  $S_t^X$  and  $S_t^Y$  are not just independent, but negatively dependent with  $\text{Cov}(S_t^X, S_t^Y) < 0$ , while keeping the distribution of each  $S_t^X$  and  $S_t^Y$  intact.

To describe the method more precisely, we employ the canonical representation of a Markov chain [30] as follows. Any given Markov chain  $X_t$  can be written as  $X_t = F(X_{t-1}, U_t)$ ,  $t \geq 1$ , where  $\{U_t : t \geq 0\}$  is a sequence of *i.i.d.* random variables uniformly distributed in  $[0, 1]$ , and the function  $F$  determines the rule of state transition. For example, suppose the state space  $\Omega$  is indexed by  $\Omega = \{1, \dots, n\}$  and let  $(\mathbf{P})_{ij} = \mathbb{P}\{X_{t+1} = j | X_t = i\}$  for  $i, j \in \Omega$ . Given  $i \in \Omega$ , set  $r(i, 0) = 0$  and  $r(i, j) = \sum_{k=1}^j (\mathbf{P})_{ik}$  for  $j \geq 1$ . Then one can construct such a function  $F : \Omega \times [0, 1) \rightarrow \Omega$  such that  $F(i, u) = j$  if  $r(i, j-1) \leq u < r(i, j)$ . A typical way [15], [31] to induce negative correlations between two Markov chains  $X_t$  and  $Y_t$  is to use a paired coupling via  $\{U_t, 1 - U_t\}$  according to

$$X_t = F(X_{t-1}, U_t), \text{ and } Y_t = F(Y_{t-1}, 1 - U_t), \quad t \geq 1.$$

This idea has been recently generalized in [18] to the cases with  $k \geq 2$  chains, where the authors utilized the concept of negative association (NA) as defined next.

**Definition 1.** [32], [18] *Real random variables,  $X_1, X_2, \dots, X_n$  are said to be negatively associated (NA) if for every pair of disjoint finite subsets  $A_1, A_2 \subset \{1, \dots, n\}$ ,*

$$\text{Cov}\{f_1(X_i, i \in A_1)f_2(X_j, j \in A_2)\} \leq 0,$$

*whenever  $f_1$  and  $f_2$  are non-decreasing (or non-increasing) in each of their arguments .*

One useful property of NA is that it is closed under independent unions [32]. Therefore, if we can generate independent vectors  $\vec{U}_t = \{U_t^1, \dots, U_t^k\}$  where each  $\vec{U}_t$ ,  $t \geq 0$  is a collection of NA random variables, then so will be the whole collection  $\{U_t^1, \dots, U_t^k, t \geq 0\}$ . Consequently,

$$\text{Cov}\{f(U_t^i, t \geq 0), f(U_t^j, t \geq 0)\} \leq 0, \quad i \neq j,$$

for any function  $f$  that is monotone (either non-increasing or non-decreasing) in each of its arguments. Taking function  $f$  to be multiple composition of the chain mapping  $X_{t+1}^i = F(X_t^i, U_{t+1}^i)$ ,  $i = 1, \dots, k$ , yields that the resulting samples from two different chains,  $i \neq j$  are NA, i.e.,  $\text{Cov}(f(X_{t_1}^i), f(X_{t_2}^j)) \leq 0$  for some  $t_1, t_2 \geq 0$ , when  $F(X, U)$  is monotone in both  $X$  and  $U$  [18]. The monotonicity with respect to  $U$  often can be followed when we use *inverse transform sampling* method, that is, generating a r.v.  $Y$  by taking  $Y = F_Y^{-1}(U)$  from  $U \sim \text{Uni}(0, 1)$  where  $F_Y$  is the cumulative distribution function (CDF). However, the monotonicity with respect to  $X$  is often very difficult to verify, especially when the state space of  $X$  is high-dimensional and there is no natural ordering relationship among states.

### B. Generation of NA random variables

An integral step to realize the antithetic coupling method is to implement a sub-module that generates negatively associated  $\{U_1, \dots, U_k\}$ , each having the same marginal distribution as the  $\text{Uni}(0, 1)$ . In case of  $k = 2$ , this step is trivial (by taking  $U$  and  $1 - U$  where  $U \sim \text{Uni}(0, 1)$ ). However, the matter is more complicated when  $k \geq 3$ .

Quantitatively, it is desirable to generate those r.v.s such that resulting samples are correlated as negatively as possible. As a measure of the degree of negative dependence, the authors in [18] used the notion of *extreme antithesis* (EA) which is defined as follows.

**Definition 2.** A set of r.v.s  $\{X_1, \dots, X_k\}$  is said to achieve extreme antithesis (EA) with respect to a marginal distribution  $\mathcal{G}$  if they are exchangeable and  $\text{Corr}(X_i, X_j) = \min\{\text{Corr}(Y_i, Y_j)\}$  where  $Y_1, \dots, Y_k$  are exchangeable, and  $\forall i, Y_i \sim \mathcal{G}$ .

It has been known that the paired coupling,  $\{U, 1 - U\}$  in case of  $k = 2$  achieves both NA and EA, however, there's no universal way of achieving both NA and EA in case of  $k \geq 3$  [17], [18]. Ref [18] provides a good survey on comparing different methods, and we here briefly revisit their pros and cons, for the completeness of our paper.

*Permuted displacement method:* First generate  $u_1 \sim \text{Uni}(0, 1)$ , and then construct

$$u_i = \{2^{i-2}u_1 + 1/2\}, i \in [2, k-1], \text{ and } u_k = 1 - \{2^{k-2}u_1\}$$

where  $\{x\} = x - [x]$  is the fractional part of  $x$ . The consequence of this method is that  $u_i$ 's are not exchangeable, nor NA in general, when  $k \geq 3$ .

*Multivariate normal method:* This method is based on the fact that negatively correlated normal r.v.s are NA [32].

- 1) Generate  $(Z_1, \dots, Z_{k-1}) \sim \text{Normal}(0, \Sigma)$  where  $(\Sigma_{ij} = -(k-1)^{-1}$  for  $i \neq j$  and  $\Sigma_{ii} = 1$ , and set  $Z_k = -(Z_1 + \dots + Z_{k-1})$ .
- 2) Compute  $U_i = \Theta(Z_i)$  for all  $i \in \{1, \dots, k\}$  where  $\Theta$  is the CDF of normal distribution  $\text{Normal}(0, 1)$ .

Due to the monotonicity of  $\Theta$ ,  $\{U_1, \dots, U_k\}$  are exchangeable and NA because  $\{Z_1, \dots, Z_k\}$  are NA. This method, however, does not achieve EA since the nonlinear transformation  $U_i = \Theta(Z_i)$  causes an additional correlation, resulting in  $\text{Corr}(U_i, U_j)$  being larger than the minimal possible value  $-(k-1)^{-1}$  [18].

### Iterative Latin Hypercube Sampling method:

- 1) Set  $\mathcal{U}_0^k = (U_0^1, \dots, U_0^k)^\top$  where  $\{U_0^{(j)}\}_{1 \leq j \leq k}$  are i.i.d.  $\text{Uni}(0, 1)$ .
- 2) For  $t = 0, 1, 2, \dots$ , find a permutation  $\mathcal{P}_t = (\varrho_t(0), \dots, \varrho_t(k-1))$  from  $\{0, \dots, k-1\}$  independently, and then set  $U_{t+1}^k = \frac{1}{k}(\varrho_t + U_t^k)$ , where  $\mathcal{U}_t^k = (U_t^1, \dots, U_t^k)^\top$  for  $t \geq 0$ .

The consequence of this method is that  $\{U_{t_1}^{(1)}, \dots, U_{t_k}^{(k)}\}$  are NA for any finite  $\{t_1, \dots, t_k\}$ , and it also practically achieves EA in the sense that  $\text{Corr}(U_t^{(i)}, U_t^{(j)}) = -\frac{1}{k-1}(1 - \frac{1}{k^{2t}})$  as the loss of EA quickly approaches to zero as  $t$  grows.

### C. Proposed algorithm: AC-CSMA

Based on the concept of antithetic coupling, we propose a new CSMA algorithm, termed *AC-CSMA*. In the proposed algorithm, we utilize the antithetically coupled random variables in deciding on-off activity at each link. An intuition behind our idea is described in the following example.

Consider a network where the delayed CSMA with  $T = 2$  is used for the scheduling algorithm, and  $\lambda_v = 1$  for all  $v \in \mathcal{N}$ . Suppose that a particular link  $v$  has sensed its channel state idle for two consecutive time slots,  $t = 0$  and  $t = 1$ , i.e.,  $\sum_{j \in \mathcal{N}(v)} \sigma_j(0) = 0$  and  $\sum_{j \in \mathcal{N}(v)} \sigma_j(1) = 0$ , and is selected in the decision schedules in these slots as illustrated in Figure 1. In the conventional delayed CSMA algorithm, the link  $v$  generates independent uniform random variables  $U_v(2)$  and  $U_v(3)$  each distributed with i.i.d.  $\text{Uni}(0, 1)$ . And then it updates its activity according to

$$\sigma_v(2) = \mathbf{1} \left\{ U_v(2) < \frac{\lambda_v}{1 + \lambda_v} \right\}, \text{ and } \sigma_v(3) = \mathbf{1} \left\{ U_v(3) < \frac{\lambda_v}{1 + \lambda_v} \right\},$$

respectively. The resulting joint probability distribution of these schedules is given in Table I-(a). Applying the antithetic coupling idea, on the other hand, if we update the link activity using negatively coupled r.v.s by  $U_v(2) \sim \text{Uni}(0, 1)$  and  $U_v(3) = 1 - U_v(2)$ , the resulting joint probability distribution changes as shown in Table I-(b), such that events  $\{\sigma_v(2) = 0, \sigma_v(3) = 0\}$  or  $\{\sigma_v(2) = 1, \sigma_v(3) = 1\}$  does not occur. As a result, the service process for the link  $v$  tends to be closer to *periodic* behavior as opposed to that with i.i.d. updates, and hence better delay performance is expected. Generalizing this idea, we propose to have each link *locally* generates  $T$  negatively associated uniform r.v.s to be used at time  $t \pmod{T} \equiv 0$ , where those r.v.s are independent across nodes.

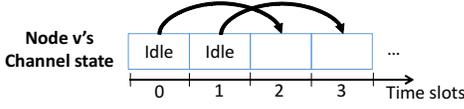


Fig. 1. An instance of channel state where link  $v$  can update its activities for the time slots  $t = 2$  and 3.

(a) Without AC		(b) With AC			
$\sigma_v(3) \backslash \sigma_v(2)$	0	1	$\sigma_v(3) \backslash \sigma_v(2)$	0	1
0	1/4	1/4	0	0	1/2
1	1/4	1/4	1	1/2	0

TABLE I

PROBABILITY DISTRIBUTION OF THE LINK  $v$ 'S ACTIVITY FOR THE TWO CONSECUTIVE TIME SLOTS,  $\sigma_v(2)$  AND  $\sigma_v(3)$ , GIVEN THAT THE LINK IS ALLOWED TO UPDATE IN THESE SLOTS.

Clearly, our algorithm does not impose any additional message overhead, and this feature is well suited to the distributed setting of wireless networks.

For more formal description of our algorithm, we write the state transition function of the CSMA algorithm described in Algorithm 1 by a function  $F : \Omega \times \Phi \times [0, 1]^n \rightarrow \Omega$  with  $\sigma(t) = F(\sigma(t-T), \mathbf{m}(t), \vec{U}(t))$ , where  $\{\mathbf{m}(t)\}_{t \geq 1}$  are *i.i.d.* random variables with probability distribution  $\alpha(\mathbf{m})$  in  $\Phi$  which determines the set of nodes to be updated by the decision schedule made at time  $t$ , and  $\vec{U}(t) \triangleq \{U_i(t)\}_{i \in \mathcal{N}}$  is a set of *i.i.d.* random variables each uniformly distributed in  $[0, 1]$  used to update transmission activity of each node. To identify an explicit expression of  $F$ , we define a *node* update function  $F_i : \Omega \times [0, 1] \rightarrow \Omega$ . More precisely, given a previous state  $\sigma \in \Omega$  and a node  $i \in \mathcal{N}$  to be updated, the next state  $\sigma' \in \Omega$  is generated by the function  $\sigma' = F_i(\sigma, u)$  given by

$$\sigma'_j = \begin{cases} \mathbf{1}\{\sum_{k \in \mathcal{N}(i)} \sigma_k = 0\} \mathbf{1}\{u < \lambda_i / (1 + \lambda_i)\}, & \text{if } j = i, \\ \sigma'_j = \sigma_j, & \text{if } j \neq i, \end{cases} \quad (2)$$

where  $u \in [0, 1]$ . Note that the decision schedule  $\mathbf{m}(t) \in \Phi$  at any time  $t$  is clearly an independent set. Suppose elements in an instance of decision schedule  $\mathbf{m}$  is sorted in an arbitrary order by  $\mathbf{m} = \{j_1, j_2, \dots, j_m\}$ . Define a *set* update function of  $\mathbf{m}$  by  $F_{[\mathbf{m}]} : \Omega \times [0, 1]^n \rightarrow \Omega$  such that

$$F_{[\mathbf{m}]}(\sigma, \vec{u}) \triangleq F_{j_m}(F_{j_{m-1}}(\dots(F_{j_1}(\sigma, u_{j_1}), \dots, u_{j_{m-1}}), u_{j_m}), \quad (3)$$

where  $\vec{u} = \{u_j\}_{j \in \mathcal{N}}$ . Then, the state transition function  $F$  can be expressed as choosing  $F_{[\mathbf{m}]}$  based on the decision schedule  $\mathbf{m}(t)$  selected at time  $t$ , i.e.,  $F(\sigma(t-T), \mathbf{m}(t), \vec{U}(t)) = F_{[\mathbf{m}(t)]}(\sigma(t-T), \vec{U}(t))$ . In our proposed algorithm, we generate  $T$  NA uniform r.v.s for each node  $i$  and for every time slot  $t \pmod T \equiv 0$ , and use them for updating its link activities for the corresponding time slots.

An additional modification we made on the delayed CSMA is that we *fix* the decision schedule for the  $T$  consecutive time slots, and the new ones are selected every other  $T$  time slots. In many practical network scenarios, the probability of a link being selected by the decision schedules is quite small, and therefore, it is often the case that links may not have a chance to fully utilize the NA property of the uniform

## Algorithm 2 AC-CSMA

- 1: Initialize: for all links  $i \in \mathcal{N}$ ,  $\sigma_i(t) = 0$ ,  $t = 0, 1, \dots, T-1$ .
- 2: At each time slot of  $t \geq T$ :
- 3: **if**  $t$  is the first time slot of every  $T$  slots, i.e.,  $t \pmod T = 0$  **then**
- 4: links find a decision schedule,  $\mathbf{m}(t)$  through a randomized procedure.
- 5: **for all** links  $i \in \mathbf{m}(t)$  **do**
- 6: Locally generate negatively associated r.v.s  $[U_i(t), \dots, U_i(t+T-1)]$ , where marginal distribution of each  $U_i(\cdot)$  is  $Uni(0, 1)$ .
- 7: **end for**
- 8: **else**  $\mathbf{m}(t) = \mathbf{m}(t-1)$
- 9: **end if**
- 10: **for all** links  $i \in \mathbf{m}(t)$  **do**
- 11: **if**  $\sum_{j \in \mathcal{N}(i)} \sigma_j(t-T) = 0$  **then**
- 12: **if**  $U_i(t) < \frac{\lambda_i}{1+\lambda_i}$  **then**  $\sigma_i(t) = 1$  **else**  $\sigma_i(t) = 0$  **end if**
- 13: **else**  $\sigma_i(t) = 0$  **end if**
- 14: **end for**
- 15: **for all** links  $j \notin \mathbf{m}(t)$  **do**
- 16:  $\sigma_i(t) = \sigma_i(t-T)$
- 17: **end for**

r.v.s within the  $T$  time slots. The rationale behind fixing the decision schedules is to maximize the impact of such negative dependence property. See Algorithm 2 for the detailed procedure of our proposed algorithm.

We note that the uniform r.v.s generated at step 6 in Algorithm 2 have the same marginal distribution as  $Uni(0, 1)$ , and they are independent across every  $T$  slots. This implies that the process  $\sigma(k), \sigma(k+T), \sigma(k+2T), \dots, \sigma(k+nT), \dots$  for each choice of  $k \in \{0, 1, \dots, T-1\}$ , evolves in the same way as that of the standard CSMA chain, thus possessing the same marginal distribution as  $\pi(\sigma)$  across all  $k$ . From this reason, our proposed algorithm also preserves the throughput-optimality. We omit this proof for brevity, and focus instead on the impact of our approach on the delay performance.

## V. ANALYSIS ON THE PERFORMANCE IMPROVEMENT

In this section, we investigate the impact of using antithetic coupling on the queueing performance. Without loss of generality, we consider a particular link  $v \in \mathcal{N}$  in the network  $\mathcal{G}$ . For ease of exposition, we denote by  $Q$  the queue size at time  $t = 0$  (or equivalently, the stationary queue size) assuming that the system has started from  $t = -\infty$ . Let  $I(t) = A(t) - \sigma(t)$  be the *net-input* into the queue at time  $t$ , with  $\mathbb{E}\{I(t)\} = \nu_v - \pi(B_v) = -\zeta < 0$ , and define  $\mathbf{I}(s, t) \triangleq \sum_{k=s}^t I(k)$ . Then, the recursion in eq. (1) admits

$$\mathbb{P}\{Q > x\} = \mathbb{P}\left\{\sup_{t \geq 0} \mathbf{I}(-t, 0) > x\right\} \quad (4)$$

Instead of directly dealing with the quantity in (4), we employ the notion of effective bandwidth that has been widely used in the queueing theory literature [22], [33], [34], [35], [36]. Conceptually, the effective bandwidth of a traffic source is related to the bandwidth needed to achieve a given QoS requirement (i.e. buffer overflow probability) when the source is offered to a constant service rate. Most of the studies in the

literature has been focused on analyzing the arrival process; however, it is straightforward to generalize it to consider both arrival and service process as a single net-input process.

Specifically, we consider the log moment generating function of the process  $\mathbf{I}(-t, 0)$  defined by

$$\Lambda_t(\theta) = \log \mathbb{E} \left[ e^{\theta \mathbf{I}(-t, 0)} \right], \quad \theta > 0. \quad (5)$$

The effective bandwidth of the net-input process can then be written as  $\Lambda_t(\theta)/(\theta t)$  [22]. Since our state-space is finite, the quantity  $\Lambda_t(\theta)$  is finite and well-defined for any given  $t, \theta$ , and thus there is one-to-one relationship between  $\Lambda_t(\theta)$  and the distribution of  $\mathbf{I}(-t, 0)$ .

While the usage of the effective bandwidth in capturing queueing performance is very rich and its full demonstration is clearly beyond the scope of this paper, the following steps epitomize the utility of  $\Lambda_t(\theta)$  in analyzing the queueing performance

$$\begin{aligned} & \log \mathbb{P} \left\{ \sup_{t \geq 0} \mathbf{I}(-t, 0) > x \right\} \approx \sup_{t \geq 0} \log \mathbb{P} \{ \mathbf{I}(-t, 0) > x \} \\ & \leq \sup_{t \geq 0} \inf_{\theta > 0} \log \mathbb{E} \left[ e^{\theta \mathbf{I}(-t, 0) - \theta x} \right] = - \inf_{t \geq 0} \sup_{\theta > 0} (\theta x - \Lambda_t(\theta)), \quad (6) \end{aligned}$$

where the first approximation is referred to as the principle of the largest term [37], and the inequality is from Chernoff bound. The term on the RHS of (6) is often called a rate function and plays a crucial role in quantifying the queueing performance. In particular, the above approximation and inequality can be made precise and becomes asymptotic equality in the large deviation theory [37], [21], [38], [22].

In what follows, we show that our proposed algorithm yields smaller  $\Lambda_t(\theta)$  than that of the delayed-CSMA algorithm, for any given  $\theta$  and  $t$ . As a byproduct, we also show that the same ordering holds between the delayed-CSMA and the standard CSMA algorithm. Note that, in view of (6), smaller  $\Lambda_t(\theta)$  means larger decay rate of  $\mathbb{P}\{Q > x\}$ , thus better queueing performance and smaller delay. We will use different superscripts 'std', 'del', and 'AC' to distinguish  $\Lambda_t(\theta)$  for the standard CSMA, delayed CSMA, and AC-CSMA, respectively. For the delayed CSMA and AC-CSMA, the same parameter  $T \geq 2$  is chosen for fair comparison in our analysis.

The major part of our analysis is to show monotonicity properties on the state transition function in (3) in order to derive a certain correlation inequality between the two algorithms in comparison. Unfortunately, we find that this is a very complicated task, as the state space  $\Omega$  forms a configurational space in  $\{0, 1\}^{|\mathcal{N}|}$ , i.e., the set of all feasible schedules (independent sets), as opposed to be a simple real domain. Even worse, it is at first sight unclear how to define an 'ordering' between two states (i.e., two feasible schedules  $\sigma_1, \sigma_2 \in \Omega$ ), a necessary component to establish the notion of monotonicity (e.g., increasing or decreasing). Nevertheless, we are able to derive the precise ordering of the moment generating function  $\Lambda_t(\theta)$  among the considered algorithms under a slightly limited setting. Specifically, we consider a situation where the conflict graph forms *bipartite*

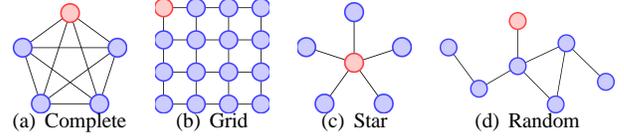


Fig. 2. Simulation topologies

in which nodes can be divided into two groups such that any neighboring node of a given node in one group belongs to the other group. For example, topologies in Figure 2-(b) and 2-(c) are bipartite, whereas those in Figure 2-(a) and 2-(d) are not.

We now state our main result in the following theorem. The entire proof involves detailed analysis, and we put it in the Appendix for brevity of presentation.

**Theorem 1.** *Under the condition that the conflict graph forms bipartite, we have*

$$\Lambda_t^{AC}(\theta) \leq \Lambda_t^{del}(\theta) \leq \Lambda_t^{std}(\theta)$$

for any given  $0 < \theta, t < \infty$ .  $\square$

Several remarks are in order. First, while we assumed that the arrival process to the queue is *i.i.d.* Bernoulli process, as typically done in the current literature, we here point out that our Theorem 1 holds true for any other general arrival process for which the moment generating function in (5) is well defined. To see this, note that (5) can be written as sum of two components, one for the arrival process and the other for the service process. Since the arrival here is assumed to be the same and independent of the service process, the ordering on  $\Lambda_t(\theta)$  implies the ordering on the same for the service process counterpart.

Second, if the net-input process is assumed to be Gaussian as in [13], part of our Theorem 1 readily recovers the main result in that paper. Specifically, if  $\mathbf{I}(-t, 0)$  is Gaussian with mean  $-\zeta t$  and variance  $v(t)$ , the ordering between  $\Lambda_t^{del}(\theta)$  and  $\Lambda_t^{std}(\theta)$  reduces to the ordering of the variance function  $v(t)$  for each  $t$ , by noting that  $\log \mathbb{E}\{e^{\theta X}\} = \theta \mu t + \theta^2 v(t)/2$  for  $X \stackrel{d}{=} \mathcal{N}(\mu t, v(t))$ . Our result here on the ordering of the moment generating function is far stronger than just a variance ordering in [13], and has natural interpretation toward the ordering of the effective bandwidth as explained earlier. In addition, Theorem 1 shows that our proposed AC-CSMA is superior than the delayed-CSMA in that it produces smaller effective bandwidth (or larger decay rate function for  $\mathbb{P}\{Q > x\}$ ) without any Gaussian assumption on the service process. Lastly, although our theoretical result holds under the stated assumptions, we again maintain that this is purely for technical purpose and our extensive simulation results in the next section confirm the same performance ordering for more general settings of network topologies.

## VI. SIMULATION RESULTS

In this section, we present numerical results for our proposed algorithm. We consider four conflict graph types: Random, Complete, Star, and Grid, as shown in Figure 2. We

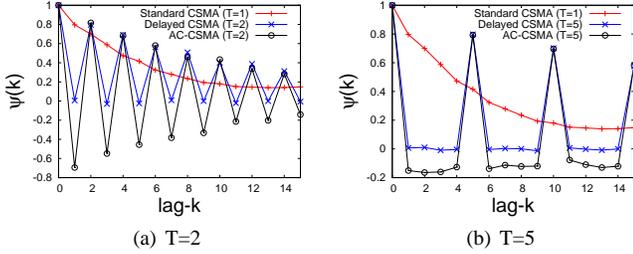


Fig. 3. Correlations of service process measured at the designated node in the Random network.

collect simulation data at a node denoted by red circle in the figure. In obtaining simulation data, we have taken average over run time of  $3 \times 10^8$  time slots, where first  $10^5$  slots of data have been discarded to remove bias from initial states. For all cases of simulation, we have fixed the access probability to 0.25 for every node.

In generating NA uniform r.v.s, we choose to use *Iterative Latin Hypercube Sampling* method since it is advantageous over the other methods discussed in Section IV-B in that it achieves both NA and EA in the steady state. With this method in use, we first look at the impact of our algorithm on correlation structure of the service process. We run simulation in Random topology with  $\lambda_i = 1.0$  for each node  $i$  in the network. We plot in Figure 3 the correlation of the service process for a range of lags under the same  $T$  in use by comparing the two algorithms: delayed CSMA and AC-CSMA. As expected, the negative correlations indeed exist at every lag that is not multiple of  $T$ .

Next, we measure the average queuing delay at the designated node for each network topology. We chose  $\lambda_v = 1.5$  for every node  $v \in \mathcal{N}$  of all simulation cases. We choose it based on our observation that 1.5 is reasonably high enough to achieve link capacities that are fairly close to a boundary of achievable capacity region. In the case of complete graph topology, for instance, each link obtains 89% of achievable capacity with  $\lambda_v = 1.5$ ,  $v \in \mathcal{N}$ . These parameters can be selected in a more sophisticated way by adopting the approach in [3]. Along with the parameter setting, we plot in Figure 4 the ratio of average delay under AC-CSMA over that under the delayed CSMA, while varying the traffic intensity with the same  $T = 2$ . Here, the traffic intensity of a link is the ratio of the average arrival rate to the average service rate of the link, i.e.,  $\nu_v / \pi(B_v) = \mathbb{E}\{A_v(t)\} / \mathbb{E}\{\sigma_v(t)\}$ . Notice that the improvement of AC-CSMA becomes larger as the system gets closer to the boundary of the capacity region, in which the induced negative dependence of the service process over time plays a bigger role.

Figure 5 shows the ratio of the average delay as before under the same traffic intensity of 0.7, while the parameter  $T$  varies from 2 to 15. The impact of  $T$  on the amount of improvement depends on the topology type and the interference relationship of the node being monitored for measurement. For instance, the improvement becomes more noticeable for larger  $T$  in Grid and Complete topology, while such difference is almost

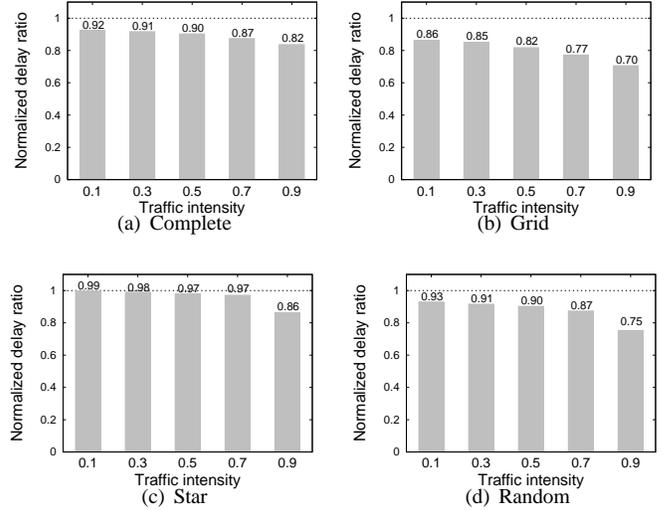


Fig. 4. Normalized ratio of average queuing delay for each topology type as traffic intensity varies. The parameter  $T = 2$  is used for delayed CSMA and AC-CSMA.

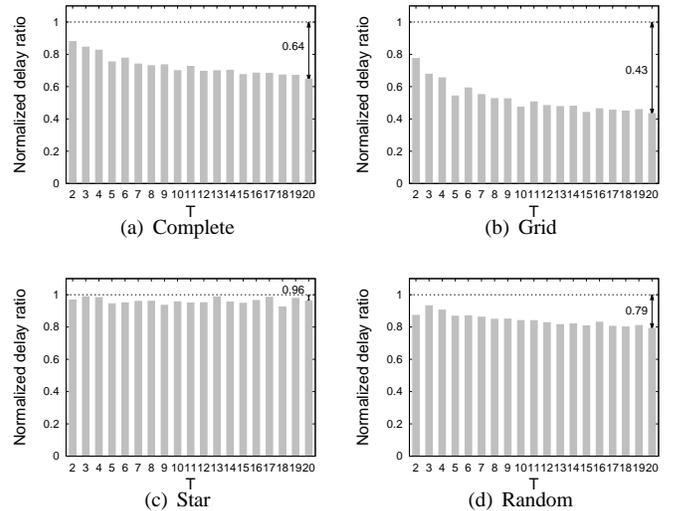


Fig. 5. Normalized ratio of average queuing delay for each topology type as  $T$  varies from 2 to 20. Here the traffic intensity is set to be 0.7.

negligible in case of Star topology. Still, in all scenarios considered, AC-CSMA clearly displays smaller delay than the delayed-CSMA, which was already known to perform better than the standard CSMA algorithm [13].

Figure 6 shows the CCDF of the queue-length in Grid and Random topologies under various choices of  $T$  and algorithms used. As expected from our Theorem 1, this tail distribution decays fastest for AC-CSMA than all others, under the same choice of  $T$ . We observed similar results in other topologies as well and omit them here due to space constraint.

## VII. CONCLUSION

We have proposed to employ the concept of antithetic coupling method that has been used to improve the accuracy

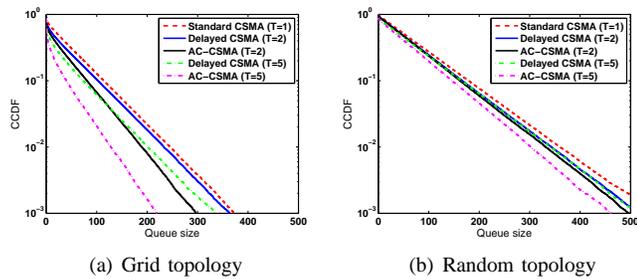


Fig. 6.  $\mathbb{P}\{Q > x\}$  in linear-log scale for (a) Grid and (b) Random topology.

of statistical estimation in parallel and multiple Monte Carlo simulations, for multi-chain based CSMA scheduling. Our proposed algorithm, AC-CSMA, can be implemented in a fully distributed manner by having each node (transmitter of a link) locally generate negatively associated random variables, to induce negative correlations in the link service process for better queueing performance. We have rigorously proved performance ordering via effective bandwidth among the standard CSMA, delayed-CSMA, and our AC-CSMA under a mild set of assumptions, and our simulation results demonstrate that the same ordering holds under much wider range of network scenarios. Our main technical contribution lies in establishing certain monotonicity for multiple instances of Glauber dynamics (or Gibbs samplers) over configurational state space with independent set constraints, for which the conventional notion of ordering no longer applies. We expect that our approach toward negatively coupled processes can also be applicable to outside of CSMA scheduling, whenever a need arises for faster simulation and computation involving multiple instances of Glauber-dynamics algorithms over highly complicated domains.

## 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] M. Leconte, J. Ni, and R. Srikant, "Improved bounds on the throughput efficiency of greedy maximal scheduling in wireless networks," in *ACM MobiHoc*, 2009.
- [3] 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.
- [4] S. Rajagopalan, D. Shah, and J. Shin, "Network adiabatic theorem: An efficient randomized protocol for contention resolution," in *ACM SIGMETRICS/Performance*, 2009.
- [5] J. Ghaderi and R. Srikant, "On the design of efficient csma algorithms for wireless networks," in *IEEE CDC*, 2010.
- [6] L. Jiang, D. Shah, J. Shin, and J. Walrand, "Distributed random access algorithm: Scheduling and congestion control," *IEEE Trans. on Information Theory*, vol. 56, pp. 6182–6207, 2010.
- [7] D. Shah and J. Shin, "Delay optimal queue-based CSMA," in *ACM SIGMETRICS*, 2010.
- [8] J. Ghaderi and R. Srikant, "Effect of access probabilities on the delay performance of q-csma algorithms," in *IEEE INFOCOM*, 2012.
- [9] M. Lotfinezhad and P. Marbach, "Throughput-optimal random access with order-optimal delay," in *IEEE INFOCOM*, 2011.
- [10] 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*, 2012.
- [11] 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*, 2012.
- [12] P.-K. Huang and X. Lin, "Improving the delay performance of CSMA algorithms: a virtual multi-channel approach," in *IEEE INFOCOM*, 2013.
- [13] J. Kwak, C.-H. Lee, and D. Y. Eun, "A high-order markov chain based scheduling algorithm for low delay in csma networks," in *IEEE INFOCOM*, 2014.
- [14] D. Lee, D. Yun, J. Shin, Y. Yi, and S.-Y. Yun, "Provable per-link delay-optimal csma for general wireless network topology," in *IEEE INFOCOM*, 2014.
- [15] J. M. Hammersley and K. W. Morton, "A new monte carlo technique: antithetic variates," *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 52, pp. 449–475, 7 1956.
- [16] J. M. Hammersley, *Monte Carlo Methods*. Springer, 2013.
- [17] R. V. Craiu and X. li Meng, "Antithetic coupling for perfect sampling," in *ISBA and Eurostat*, 2001.
- [18] R. V. Craiu and X.-L. Meng, "Multiprocess parallel antithetic coupling for backward and forward markov chain monte carlo," *Annals of Statistics*, vol. 33, no. 2, pp. 661–697, 2005.
- [19] L. Jiang and J. Walrand, "A distributed csma algorithm for throughput and utility maximization in wireless networks," *IEEE/ACM Trans. on Networking*, vol. 18, pp. 960–972, 2010.
- [20] R. G. Addie and M. Zukerman, "An Approximation for Performance Evaluation of Stationary Single Server Queues," *IEEE Trans. on Communications*, vol. 42, no. 12, pp. 3150–3160, Dec. 1994.
- [21] 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.
- [22] F. Kelly, "Notes on effective bandwidths," *Stochastic networks: Theory and Applications*, Oxford University Press, 1996.
- [23] C. J. Geyer, "Practical Markov Chain Monte Carlo," *Statistical Science*, vol. 7, no. 4, pp. 437–483, Nov. 1992.
- [24] J. S. Liu, W. H. Wong, and A. Kong, "Covariance structure and convergence rate of the gibbs sampler with various scans," *Journal of the Royal Statistical Society, Ser. B.*, vol. 57, pp. 157–169, 1995.
- [25] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov chains and mixing times*. American Mathematical Society, 2009.
- [26] 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," in *IEEE INFOCOM*, 2010.
- [27] 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.
- [28] P. Brémaud, *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*. Springer-Verlag, 1999.
- [29] X. Xu, C.-H. Lee, and D. Y. Eun, "A general framework of hybrid graph sampling for complex network analysis," in *IEEE INFOCOM*, 2014.
- [30] F. Bacelli and P. Brémaud, *Elements of Queueing Theory: Palm Martingale Calculus and Stochastic Recurrences*. Springer-Verlag, 2003.
- [31] A. Frigessi, J. Gasemyr, and H. Hue, "Antithetic coupling of two gibbs sampler chains," *Annals of Statistics*, vol. 28, no. 4, 2000.
- [32] K. Joag-Dev and F. Proschan, "Negative association of random variables with applications," *Annals of Statistics*, vol. 11, no. 1, pp. 286–295, 1983.
- [33] F. Kelly, "Effective bandwidths at multi-class queues," *Queueing Systems*, vol. 9, no. 1-2, pp. 5–15, 1991.
- [34] A. W. Berger and W. Whitt, "Effective bandwidths with priorities," *IEEE/ACM Trans. on Networking*, vol. 6, pp. 447–460, August 1998.
- [35] R. J. Gibbens, "Traffic characterisation and effective bandwidths for broadband network traces," 1996.
- [36] G. Kesidis, J. Walrand, and C.-S. Chang, "Effective bandwidths for multiclass markov fluids and other atm sources," *IEEE/ACM Trans. on Networking*, vol. 1, no. 4, pp. 424–428, Aug 1993.
- [37] A. Ganesh, N. O'Connell, and D. Wischik, *Big Queues*. Springer, 2004.
- [38] P. W. Glynn and W. Whitt, "Logarithmic asymptotics for steady-state tail probabilities in a single-server queue," *Journal of Applied Probability*, vol. 31, pp. 131–156, 1994.
- [39] C. M. Fortuin and P. W. Kasteleyn, "Correlation inequalities on some partially ordered sets," *Commun. math. Phys.*, vol. 22, pp. 89–103, 1971.

## APPENDIX

## A. Proof of Proposition 1

*Proof.* We denote by  $X_t$  the schedule instance at time  $t$  that consists of activity state of  $n$  nodes in the network. We use a notation  $X_t^i$  to denote the node  $i$ 's state from  $X_t$ . For a given schedule instance  $X_t$ , we denote by  $X_t^{[J]} \triangleq \{X_t^i : i \in J\}$  its subset state for  $J \subset \mathcal{N}$ . We define  $X_t^{[-J]} \triangleq \{X_t^i : i \in \mathcal{N} \setminus J\}$  that of subset excluding nodes in  $J$ . In view of probability distribution, we use  $\mathbb{P}\{X = x\}$  for  $x \in \Omega$  to imply  $\mathbb{P}\{X^1 = x^1, \dots, X^n = x^n\}$  where  $x^i \in \{0, 1\}, \forall i \in \mathcal{N}$ . For a given joint distribution  $\mathbb{P}\{X = x\}$ , one can find a well-defined marginal distribution  $\mathbb{P}\{X^{[J]} = x^{[J]}\}$  for any subset  $J \subset \mathcal{N}$ . More precisely, the marginal distribution can be obtained by

$$\mathbb{P}\{X^{[J]} = x^{[J]}\} = \sum_{y \in \Omega(x^{[J]})} \mathbb{P}\{X = y\}, \quad (7)$$

where  $\Omega(x^{[J]}) = \{y \in \Omega : x^i = y^i, \forall i \in J\}$ .

Without loss of generality, we prove the lemma by showing  $\mathbb{E}\{f(X_t)f(X_{t+1})\} \geq 0$  for any function  $f$ . The statement in lemma can be established from it by choosing function  $g(\cdot) = f(\cdot) - \mathbb{E}\{f(\cdot)\}$ , and noticing  $\mathbb{E}\{g(X_t)g(X_{t+1})\} = \mathbb{E}\{f(X_t)f(X_{t+1})\} - \mathbb{E}\{f(X_t)\}^2 \geq 0$ .

At every time  $t$ , the network selects an independent set  $\mathbf{m} \in \Phi$  as a decision schedule  $\mathbf{M}_t$  with a probability distribution  $\alpha(\mathbf{m})$ . From the algorithm, we can derive

$$\begin{aligned} & \mathbb{E}\{f(X_t)f(X_{t+1})\} \\ &= \mathbb{E}\{\mathbb{E}\{f(X_t)f(X_{t+1})|\mathbf{M}_{t+1}\}\} \\ &= \mathbb{E}\{\mathbb{E}\{\mathbb{E}\{f(X_t)f(X_{t+1})|\mathbf{M}_{t+1}, X_t^{[-\mathbf{M}_{t+1}]}\}|\mathbf{M}_{t+1}\}\} \\ &= \sum_{\mathbf{m} \in \Phi} \alpha(\mathbf{m}) \mathbb{E}\{\mathbb{E}\{f(X_t)f(X_{t+1})|X_t^{[-\mathbf{m}]}\}\}. \end{aligned} \quad (8)$$

Since each node  $i$  selected by the decision schedule  $\mathbf{M}(t+1) = \mathbf{m}$  at time  $t+1$ , updates its channel state  $X_{t+1}^i$  irrespective of those states  $X_t^j, \forall j \notin \mathbf{m}$  including its own previous state  $X_t^i$ ,  $X_t$  and  $X_{t+1}$  are conditionally independent, i.e.,

$$\begin{aligned} & \mathbb{E}\{f(X_t)f(X_{t+1})|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} \\ &= \mathbb{E}\{f(X_t)|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} \mathbb{E}\{f(X_{t+1})|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} \end{aligned} \quad (9)$$

The proof completes by showing that those two conditional expectations are identical. To see this, observe that under stationarity,  $\mathbb{P}\{X_t = x\} = \pi(x)$ , and  $\mathbb{P}\{X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} = \pi(x^{[-\mathbf{m}]})$ , where

$$\begin{aligned} \pi(x^{[-\mathbf{m}]}) &= \frac{1}{Z} \sum_{y \in \Omega(x^{[-\mathbf{m}]})} \prod_{j \in \mathcal{N} \setminus \mathbf{m}} \lambda_j^{y^j} \prod_{k \in \mathbf{m}} \lambda_k^{y^k} \\ &= \frac{1}{Z} \prod_{j \in \mathcal{N} \setminus \mathbf{m}} \lambda_j^{x^j} \prod_{k \in \mathbf{m}} (1 + \lambda_k) \end{aligned}$$

which is obtained from the equation (7). From the definition

of conditional probability, we have

$$\begin{aligned} \mathbb{P}\{X_t = y|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} &= \frac{\mathbb{P}\{X_t = y, X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\}}{\mathbb{P}\{X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\}} \\ &= \begin{cases} \frac{\pi(y)}{\pi(x^{[-\mathbf{m}]})} & \text{if } x^i = y^i, \forall i \notin \mathbf{m}, \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

This implies

$$\begin{aligned} \mathbb{E}\{f(X_t)|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} &= \sum_{y \in \Omega} f(y) \mathbb{P}\{X_t = y|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} \\ &= \sum_{y \in \Omega(x^{[-\mathbf{m}]})} f(y) \frac{\pi(y)}{\pi(x^{[-\mathbf{m}]})} = \sum_{y \in \Omega(x^{[-\mathbf{m}]})} f(y) \left( \prod_{i \in \mathbf{m}} \frac{\lambda_i^{y^i}}{1 + \lambda_i} \right). \end{aligned}$$

According to the rule of algorithm, the links not selected by the decision schedule  $\mathbf{m}$  cannot change their activity state, and therefore it must be  $X_{t+1}^i = X_t^i$  for  $i \notin \mathbf{m}$ . Then we have,

$$\begin{aligned} & \mathbb{P}\{X_{t+1} = y|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} \\ &= \begin{cases} \mathbb{P}\{X_{t+1}^{[\mathbf{m}]} = y^{[\mathbf{m}]}|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} & \text{if } x^i = y^i, \forall i \notin \mathbf{m}, \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

We can rewrite the conditional expectation for the lag one term in the equation (9) as follows,

$$\begin{aligned} & \mathbb{E}\{f(X_{t+1})|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} \\ &= \sum_{y \in \Omega} f(y) \mathbb{P}\{X_{t+1} = y|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} \\ &= \sum_{y \in \Omega(x^{[-\mathbf{m}]})} f(y) \mathbb{P}\{X_{t+1}^{[\mathbf{m}]} = y^{[\mathbf{m}]}|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} \\ &= \sum_{y \in \Omega(x^{[-\mathbf{m}]})} f(y) \left( \prod_{i \in \mathbf{m}} \frac{\lambda_i^{y^i}}{1 + \lambda_i} \right), \end{aligned}$$

where the last equality is from the state update rule for each link  $i \in \mathbf{m}$  in the algorithm. Clearly, it turns out,

$$\mathbb{E}\{f(X_t)|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\} = \mathbb{E}\{f(X_{t+1})|X_t^{[-\mathbf{m}]} = x^{[-\mathbf{m}]}\},$$

under stationarity, and hence the equation (8) is equivalent to the following,

$$\sum_{\mathbf{m} \in \Phi} \alpha(\mathbf{m}) \mathbb{E}\{\mathbb{E}^2\{f(X_{t+1})|X_t^{[-\mathbf{m}]}\}\} \geq 0$$

which completes the proof.  $\square$

## B. Proof of Theorem 1

*Proof.* According to the system model we described in Section II-A, the packet arrival process of each link follows Bernoulli process. Due to the fact that a link's packet arrival process is independent from its service process, it holds  $\mathbb{E}[e^{\theta \mathbf{I}(-t, 0)}] = \mathbb{E}[e^{\theta \sum_{i=-t}^0 A(i)}] \mathbb{E}[e^{-\theta \sum_{i=-t}^0 \sigma(i)}]$ . Therefore, we only need to show an ordering relationship in terms of  $\mathbb{E}[e^{-\theta \sum_{i=-t}^0 \sigma(i)}]$ . Define by  $S(t) = \sum_{i=-t}^0 \sigma_v(i)$  the cumulative service process of link  $v$ , and use different superscripts 'std', 'del', and 'AC' to distinguish  $S(t)$  for the standard CSMA, delayed CSMA, and AC-CSMA, respectively. In the later part of our analysis,

we will show in separate,  $\mathbb{E}[e^{-\theta S^{\text{std}}(t)}] \geq \mathbb{E}[e^{-\theta S^{\text{del}}(t)}]$  and  $\mathbb{E}[e^{-\theta S^{\text{del}}(t)}] \geq \mathbb{E}[e^{-\theta S^{\text{AC}}(t)}]$  to complete the proof.

Before presenting the main parts of our proof, we collect some definitions and notations that will be used throughout the proof. For a given graph  $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ , we define by  $\Omega$  the set of all feasible schedules. According to the bipartite graph condition, we divide the set of links  $\mathcal{N}$  into two groups,  $\mathcal{N}^+$  and  $\mathcal{N}^-$ , such that any neighboring node  $j$  of a node  $i \in \mathcal{N}^+$  (or  $i \in \mathcal{N}^-$ ) belongs to the other group,  $\mathcal{N}^-$  (or  $\mathcal{N}^+$ ). Without loss of generality, we assume that the link  $v$  of our interest belongs to the group  $\mathcal{N}^+$ .

The subsequent parts of our proof rely on constructing a certain ordering relationship over the set of feasible schedules. For this purpose, we define by ' $\preceq_\Omega$ ' a partial ordering on  $\Omega$ , but with partially *turned-around* ordering relation, such that two states  $\sigma, \sigma' \in \Omega$  satisfy  $\sigma \preceq \sigma'$  if and only if  $\sigma_i \leq \sigma'_i$  for  $i \in \mathcal{N}^+$  and  $\sigma_j \geq \sigma'_j$  for  $j \in \mathcal{N}^-$ . Also, for a vector of real variable  $\mathbf{u} = (u_i)_{i \in \mathcal{N}} \in [0, 1]^{|\mathcal{N}|}$ , where each  $u_i \in [0, 1]$  is associated with node  $i$  to update its link state (according to (2)), we define similarly by ' $\preceq_{\mathbb{R}}$ ' on  $[0, 1]^{|\mathcal{N}|}$  such that two vectors  $\mathbf{u}, \mathbf{u}' \in [0, 1]^{|\mathcal{N}|}$  satisfies  $\mathbf{u} \preceq_{\mathbb{R}} \mathbf{u}'$  if and only if  $u_i \leq u'_i$  for all  $i \in \mathcal{N}_+$  and  $u_j \geq u'_j$  for all  $j \in \mathcal{N}^-$ . Using this notation, we first obtain several sub-lemmas, that are useful to show our main result.

**Lemma 1.** *The set-update function  $F_{[\mathbf{m}]}(\sigma, \mathbf{u})$  in eq. (3) satisfies the following.*

$$\sigma \preceq_\Omega \sigma' \Rightarrow F_{[\mathbf{m}]}(\sigma, \mathbf{u}) \preceq_\Omega F_{[\mathbf{m}]}(\sigma', \mathbf{u}), \quad \forall \mathbf{u} \in [0, 1]^{|\mathcal{N}|}, \quad (10)$$

for any  $\sigma, \sigma' \in \Omega$ ,  $\mathbf{m} \in \Phi$ , and  $\mathbf{u} \in [0, 1]^{|\mathcal{N}|}$ , and,

$$\mathbf{u} \preceq_{\mathbb{R}} \mathbf{u}' \Rightarrow F_{[\mathbf{m}]}(\sigma, \mathbf{u}) \preceq_\Omega F_{[\mathbf{m}]}(\sigma, \mathbf{u}'), \quad \forall \sigma \in \Omega, \quad (11)$$

for any  $\sigma \in \Omega$ ,  $\mathbf{m} \in \Phi$ , and  $\mathbf{u}, \mathbf{u}' \in [0, 1]^{|\mathcal{N}|}$ .

*Proof.* Consider any node  $i \in \mathcal{N}$ , and observe the following derivation.

$$\begin{aligned} & \sigma \preceq_\Omega \sigma' \\ & \Rightarrow \begin{cases} \sigma_i \leq \sigma'_i, \sigma_j \geq \sigma'_j, \forall j \in N(i), & \text{if } i \in \mathcal{N}^+, \\ \sigma_i \geq \sigma'_i, \sigma_j \leq \sigma'_j, \forall j \in N(i), & \text{if } i \in \mathcal{N}^- \end{cases} \\ & \Rightarrow \begin{cases} \mathbf{1}\{\sum_{j \in N(i)} \sigma_j = 0\} \leq \mathbf{1}\{\sum_{j \in N(i)} \sigma'_j = 0\} & \text{if } i \in \mathcal{N}^+, \\ \mathbf{1}\{\sum_{j \in N(i)} \sigma_j = 0\} \geq \mathbf{1}\{\sum_{j \in N(i)} \sigma'_j = 0\} & \text{if } i \in \mathcal{N}^- \end{cases} \\ & \Rightarrow \begin{cases} (F_i(\sigma, \mathbf{u}))_i \leq (F_i(\sigma', \mathbf{u}))_i, & \text{if } i \in \mathcal{N}^+, \\ (F_i(\sigma, \mathbf{u}))_i \geq (F_i(\sigma', \mathbf{u}))_i, & \text{if } i \in \mathcal{N}^- \end{cases} \\ & \Rightarrow F_i(\sigma, \mathbf{u}) \preceq_\Omega F_i(\sigma', \mathbf{u}). \end{aligned}$$

Since  $F_{[\mathbf{m}]}$  is simply a multiple composition of  $(F_i)_{i \in \mathbf{m}}$  in an arbitrarily ordered set  $\mathbf{m}$ , as defined in eq. (3), if  $\sigma \preceq_\Omega \sigma'$  holds, then it follows  $F_{[\mathbf{m}]}(\sigma, \mathbf{u}) \preceq_\Omega F_{[\mathbf{m}]}(\sigma', \mathbf{u})$  for any given  $\mathbf{m} \in \Phi$  and  $\mathbf{u} \in [0, 1]^{|\mathcal{N}|}$ , which verifies eq. (10).

And also, note that

$$\mathbf{u} \preceq_{\mathbb{R}} \mathbf{u}'$$

$$\begin{aligned} & \Rightarrow \begin{cases} u_i \leq u'_i \Rightarrow (F_i(\sigma, u_i))_i \leq (F_i(\sigma, u'_i))_i, & \text{if } i \in \mathcal{N}^+ \\ u_i \geq u'_i \Rightarrow (F_i(\sigma, u_i))_i \geq (F_i(\sigma, u'_i))_i, & \text{if } i \in \mathcal{N}^- \end{cases} \\ & \Rightarrow F_i(\sigma, \mathbf{u}) \preceq_\Omega F_i(\sigma, \mathbf{u}'). \end{aligned}$$

Similarly, due to the fact that  $F_{[\mathbf{m}]}$  is a multiple composition of  $F_i$ ,  $i \in \mathbf{m}$ , it naturally follows  $F_{[\mathbf{m}]}(\sigma, \mathbf{u}) \preceq_\Omega F_{[\mathbf{m}]}(\sigma, \mathbf{u}')$  for any given  $\mathbf{m} \in \Phi$  and  $\sigma \in \Omega$  as given by eq. (11).  $\square$

We here describe some terminologies from lattice theory. A partially ordered set  $\Xi = (\Omega_L, \preceq_L)$  is called *lattice* if any two elements  $x$  and  $y$  in a set  $\Omega_L$  have a least upper bound  $x \vee y$  and a greatest lower bound  $x \wedge y$  in a partial order  $\preceq_L$ . And, a lattice is called *distributive* if the operations  $\vee$  and  $\wedge$  satisfy either of the following two equivalent conditions,

$$\begin{aligned} x \wedge (y \vee z) &= (x \wedge y) \vee (x \wedge z) \quad \text{for all } x, y, z \in \Omega_L, \\ x \vee (y \wedge z) &= (x \vee y) \wedge (x \vee z) \quad \text{for all } x, y, z \in \Omega_L. \end{aligned}$$

A real function  $f : \Omega_L \rightarrow \mathbb{R}$  on a partially ordered set  $\Xi$  is called *increasing* if it holds  $f(x) \leq f(y)$  for any ordered pair  $x \preceq_L y$  of elements in  $\Xi$ . The following is a well known correlation inequality theorem called *FKG-theorem* [39].

**Lemma 2.** [39] *Let  $\Xi$  be a finite distributive lattice, and  $\mu$  a nonnegative function on  $\Omega_L$ , that satisfies the log-supermodularity condition, i.e.,*

$$\mu(x \wedge y)\mu(x \vee y) \geq \mu(x)\mu(y)$$

for all  $x, y$  in the lattice  $\Xi$ . Then, for increasing functions  $f_1$  and  $f_2$ , we have

$$\langle f_1 f_2 \rangle \geq \langle f_1 \rangle \langle f_2 \rangle$$

where  $\langle f \rangle \triangleq (\sum_{x \in \Omega} f(x)\mu(x)) / (\sum_{x \in \Omega} \mu(x))$ . Also, the same inequality holds if both  $f_1$  and  $f_2$  are decreasing.

**Lemma 3.** *Let  $X$  be a random variable over the state space  $\Omega$  with  $\mathbb{P}\{X = x\} = \pi(x)$ . For all increasing functions (or all decreasing functions)  $g_1$  and  $g_2$ , we have*

$$\mathbb{E}\{g_1(X)g_2(X)\} \geq \mathbb{E}\{g_1(X)\}\mathbb{E}\{g_2(X)\},$$

where  $g_i : \Omega \rightarrow \mathbb{R}$ ,  $i = 1, 2$  is called *increasing* if  $g_i(x) \leq g_i(x')$  (decreasing if  $g_i(x) \geq g_i(x')$ ) holds for any  $x \preceq_\Omega x'$ .

*Proof.* Considering a partial order set  $(\Omega, \preceq_\Omega)$ , we first show that it is lattice. To see this, we need to check for any two states  $x, y \in \Omega$ , both  $x \vee y$  and  $x \wedge y$  are also elements in  $\Omega$ . Due to the hard core constraint of the feasible schedules, for any element  $x \in \Omega$ , it satisfies  $\sum_{j \in N(i)} x_j = 0$  if  $x_i = 1$  for all node  $i \in \mathcal{N}$ . Therefore, the above condition can be checked by showing that for any two states  $x, y \in \Omega$  and any node  $i \in \mathcal{N}$ , it holds  $\sum_{j \in N(i)} (x \vee y)_j = 0$  when  $(x \vee y)_i = 1$ , and  $\sum_{j \in N(i)} (x \wedge y)_j = 0$  when  $(x \wedge y)_i = 1$ . According to the way how we defined the partial ordering, for any two states  $x, y \in \Omega$ , we have

$$(x \vee y)_i = \begin{cases} \max(x_i, y_i) & \text{if } i \in \mathcal{N}^+ \\ \min(x_i, y_i) & \text{if } i \in \mathcal{N}^-, \end{cases} \quad (12)$$

$$(x \wedge y)_i = \begin{cases} \min(x_i, y_i) & \text{if } i \in \mathcal{N}^+ \\ \max(x_i, y_i) & \text{if } i \in \mathcal{N}^- \end{cases} \quad (13)$$

Consider any arbitrary two states  $x, y \in \Omega$  and a node  $i \in \mathcal{N}^+$ . If  $(x \vee y)_i = 1$ , then either  $x_i = 1$  or  $y_i = 1$  due to eq. (12). This implies either  $x_j = 0, \forall j \in \mathcal{N}(i)$  or  $y_j = 0, \forall j \in \mathcal{N}(i)$ . Since all the neighboring nodes  $j \in \mathcal{N}(i)$  belong to  $\mathcal{N}^-$ , one can check  $\sum_{j \in \mathcal{N}(i)} (x \vee y)_j = \sum_{j \in \mathcal{N}(i)} \min(x_j, y_j) = 0$ . If  $(x \wedge y)_i = 1$  on the other hand, both  $x_i = 1$  and  $y_i = 1$  due to eq. (13). This implies all the  $x_j = y_j = 0, \forall j \in \mathcal{N}(i)$ , and thus clearly,  $\sum_{j \in \mathcal{N}(i)} (x \vee y)_j = 0$ .

Similarly, consider two states  $x, y \in \Omega$  and a node  $i \in \mathcal{N}^-$ . If  $(x \vee y)_i = 1$ , then  $x_i = y_i = 1$ , which in turn implies  $x_j = y_j = 0, \forall j \in \mathcal{N}(i)$ . Since all the neighboring nodes  $j \in \mathcal{N}(i)$  belong to  $\mathcal{N}^+$ , the condition implies  $\sum_{j \in \mathcal{N}(i)} (x \vee y)_j = 0$ . If  $(x \wedge y)_i = 1$ , then either  $x_i = 1$  or  $y_i = 1$ . This implies either  $x_j = 0, \forall j \in \mathcal{N}(i)$  or  $y_j = 0, \forall j \in \mathcal{N}(i)$ . Clearly, this in turn implies  $\sum_{j \in \mathcal{N}(i)} (x \vee y)_j = \min(x_j, y_j) = 0$ .

In addition to the lattice condition, we check that  $(\Omega, \preceq_\Omega)$  is also distributive by noticing,

$$\begin{aligned} & (x \wedge (y \vee z))_i \\ &= \begin{cases} \min(x_i, \max(y_i, z_i)) = \min(\max(x_i, y_i), \max(x_i, z_i)), & \text{if } i \in \mathcal{N}^+ \\ \max(x_i, \min(y_i, z_i)) = \max(\min(x_i, y_i), \min(x_i, z_i)), & \text{if } i \in \mathcal{N}^- \end{cases} \\ &= ((x \wedge y) \vee (x \wedge z))_i. \end{aligned}$$

We view the non-negative function  $\mu$  in Lemma 2, as a probability distribution  $\mu(x) = \pi(x)$  for  $x \in \Omega$ . See that

$$\begin{aligned} \pi(x \vee y) &= \frac{1}{Z} \prod_{i \in \mathcal{N}^+} \lambda^{\max(x_i, y_i)} \prod_{j \in \mathcal{N}^-} \lambda^{\min(x_j, y_j)}, \\ \pi(x \wedge y) &= \frac{1}{Z} \prod_{i \in \mathcal{N}^+} \lambda^{\min(x_i, y_i)} \prod_{j \in \mathcal{N}^-} \lambda^{\max(x_j, y_j)}, \end{aligned}$$

and,

$$\begin{aligned} & \pi(x \vee y) \pi(x \wedge y) \\ &= \frac{1}{Z^2} \prod_{i \in \mathcal{N}^+} \lambda^{\max(x_i, y_i) + \min(x_i, y_i)} \prod_{j \in \mathcal{N}^-} \lambda^{\max(x_j, y_j) + \min(x_j, y_j)} \\ &= \frac{1}{Z^2} \prod_{i \in \mathcal{N}} \lambda^{x_i + y_i} = \pi(x) \pi(y), \end{aligned}$$

which implies that  $\pi$  satisfies log-supermodularity condition. By using Lemma 2 and observing that the definition of increasing functions  $g_1$  and  $g_2$  are in accordance with the functions  $f_1$  and  $f_2$  defined in Lemma, we obtain the result.  $\square$

**Proof of  $\mathbb{E}[e^{-\theta S^{\text{sid}}(t)}] \geq \mathbb{E}[e^{-\theta S^{\text{del}}(t)}]$ :** We use  $\sigma^{(T)}(s)$  to denote the schedule at time  $s$  governed by the delayed CSMA algorithm with parameter  $T$ . The schedule with  $T = 1$ ,  $\sigma^{(1)}(s)$ , is the one by the standard CSMA, for which we will drop the superscript and simply denote it by  $\sigma(s)$ .

**Lemma 4.** For given  $s, k$  where  $0 < s < k < \infty$ , define two disjoint subsets  $A_1$  and  $A_2$  such that  $A_1 = [0, \dots, s]$  and  $A_2 = [s+1, \dots, k]$ . For all increasing (or all decreasing) functions  $h_i, i = 1, 2$ , each taking arguments from  $\sigma(A_1) \triangleq$

$\{\sigma(0), \dots, \sigma(s)\}$  and  $\sigma(A_2) \triangleq \{\sigma(s+1), \dots, \sigma(k)\}$ , respectively, we have

$$\mathbb{E}\{h_1(\sigma(A_1))h_2(\sigma(A_2))\} \geq \mathbb{E}\{h_1(\sigma(A_1))\}\mathbb{E}\{h_2(\sigma(A_2))\} \quad (14)$$

where  $h_i : \Omega^{|A_i|} \rightarrow \mathbb{R}, i = 1, 2$ , is increasing if  $h_i(\sigma(A_i)) \leq h_i(\sigma'(A_i))$  (decreasing if  $h_i(\sigma(A_i)) \geq h_i(\sigma'(A_i))$ ) whenever  $\sigma(s) \preceq_\Omega \sigma'(s)$  for all  $s \in A_i$ , under the condition that  $\mathbb{P}\{\sigma(0) = x\} = \pi(x)$ , for any  $x \in \Omega$ .

*Proof.* Denote by  $P_{x,y}$  the transition probability of the state from  $x$  to  $y$ , where  $x, y \in \Omega$ , due to the standard CSMA algorithm. And, observe that

$$\begin{aligned} & \mathbb{E}\{h_1(\sigma(0), \dots, \sigma(s))h_2(\sigma(s+1), \dots, \sigma(k))\} \\ &= \sum_{x_0, \dots, x_k \in \Omega} \pi(x_0) \prod_{j=0}^{k-1} P_{x_j, x_{j+1}} h_1(x_0, \dots, x_s) h_2(x_{s+1}, \dots, x_k) \\ &= \sum_{x_0, \dots, x_k \in \Omega} \pi(x_0) \prod_{j=0}^{s-1} P_{x_j, x_{j+1}} h_1(x_0, \dots, x_s) \prod_{j=s}^{k-1} P_{x_j, x_{j+1}} h_2(x_{s+1}, \dots, x_k) \\ &\stackrel{(a)}{=} \sum_{x_0, \dots, x_k \in \Omega} \pi(x_s) \prod_{j=0}^{s-1} P_{x_{j+1}, x_j} h_1(x_0, \dots, x_s) \prod_{j=s}^{k-1} P_{x_j, x_{j+1}} h_2(x_{s+1}, \dots, x_k) \\ &= \mathbb{E}\{\mathbb{E}\{h_1(\tilde{\sigma}(0), \dots, \tilde{\sigma}(s))h_2(\sigma(s+1), \sigma(k)) | \sigma(s)\}\} \end{aligned} \quad (15)$$

where (a) is due to the time reversibility, and the transition probability for  $\tilde{\sigma}$  in the last equation is defined by

$$\mathbb{P}\{\tilde{\sigma}(j) = y | \tilde{\sigma}(j+1) = x\} = P_{x,y}, \quad j = 0, \dots, s-1,$$

for  $x, y \in \Omega$ , and  $\tilde{\sigma}(s) = \sigma(s)$ . Since the tuples of random variables  $\{\tilde{\sigma}(0), \dots, \tilde{\sigma}(s)\}$  and  $\{\sigma(s+1), \dots, \sigma(k)\}$  are conditionally independent given  $\sigma(s)$ , we have

$$\begin{aligned} (15) &= \sum_{x \in \Omega} \pi(x) \mathbb{E}\{h_1(\tilde{\sigma}(0), \dots, \tilde{\sigma}(s)) | \tilde{\sigma}(s) = x\} \\ &\quad \cdot \mathbb{E}\{h_2(\sigma(s+1), \dots, \sigma(k)) | \sigma(s) = x\} \end{aligned} \quad (16)$$

Define by  $\mathcal{F}_1 \triangleq \{\mathbf{m}(1), \dots, \mathbf{m}(s), \bar{U}(1), \dots, \bar{U}(s)\}$  and  $\mathcal{F}_2 \triangleq \{\mathbf{m}(s+1), \dots, \mathbf{m}(k), \bar{U}(s+1), \dots, \bar{U}(k)\}$ . Since all the random variables,  $\mathbf{m}(1), \dots, \mathbf{m}(k)$  and  $\bar{U}(1), \dots, \bar{U}(k)$  are independent from any other events, we have

$$\begin{aligned} & \mathbb{E}\{h_1(\tilde{\sigma}(0), \dots, \tilde{\sigma}(s)) | \tilde{\sigma}(s) = x\} \\ &= \mathbb{E}\{\mathbb{E}\{h_1(\tilde{\sigma}(0), \dots, \tilde{\sigma}(s)) | \tilde{\sigma}(s) = x, \mathcal{F}_1\}, \text{ and} \\ & \mathbb{E}\{h_2(\sigma(s+1), \dots, \sigma(k)) | \sigma(s) = x\} \\ &= \mathbb{E}\{\mathbb{E}\{h_2(\sigma(0), \dots, \sigma(s)) | \sigma(s) = x, \mathcal{F}_2\}. \end{aligned}$$

Define the following events,

$$\begin{aligned} E_1 &\triangleq \{\mathbf{m}(1) = m_1, \dots, \mathbf{m}(s) = m_s, \bar{U}(1) = \mathbf{u}_1, \dots, \bar{U}(s) = \mathbf{u}_s\}, \\ E_2 &\triangleq \{\mathbf{m}(s+1) = m_{s+1}, \dots, \mathbf{m}(k) = m_k, \bar{U}(s+1) = \mathbf{u}_{s+1}, \dots, \bar{U}(k) = \mathbf{u}_k\}. \end{aligned}$$

Under the event  $E_1$ , we can write  $\tilde{\sigma}(j) = F_{[m_{j+1}]}(\tilde{\sigma}(j+1), \mathbf{u}_{j+1})$ , for  $j = 0, \dots, s-1$ . According to the Lemma 1,  $\tilde{\sigma}(s-1) = F_{[m_s]}(\tilde{\sigma}(s), \mathbf{u}_s)$  is increasing in  $\mathbf{u}_s$  in the sense that if  $\mathbf{u}_s \preceq_{\mathbb{R}} \mathbf{u}'_s$  then  $F_{[m_s]}(\tilde{\sigma}(s), \mathbf{u}_s) \preceq_{\Omega} F_{[m_s]}(\tilde{\sigma}(s), \mathbf{u}'_s)$ ; The state update function,  $F_{[m_{s-1}]}(F_{[m_s]}(\tilde{\sigma}(s), \mathbf{u}_s), \mathbf{u}_{s-1})$  for generating  $\tilde{\sigma}(s-2)$  is also increasing in  $\mathbf{u}_s, \mathbf{u}_{s-1}$ ; and so

on. This indicates that  $h'_1(x) \triangleq \mathbb{E}\{h_1(\tilde{\sigma}(0), \dots, \tilde{\sigma}(s)) | \tilde{\sigma}(s) = x, E_1\}$  is increasing function of  $x$  when so is  $h_2$ , in the sense that if two states  $x, x' \in \Omega$  satisfies  $x \preceq_{\Omega} x'$  then  $h'_1(x) \leq h'_1(x')$  holds. Similarly, under the event  $E_2$ , we can write  $\sigma(j+1) = F_{[m_{j+1}]}(\sigma(j), \mathbf{u}_{j+1})$ , for  $j = s, \dots, k-1$ , and verify that  $h'_2(x) \triangleq \mathbb{E}\{h_2(\sigma(s+1), \dots, \sigma(k)) | \sigma(s) = x, E_2\}$  is increasing in  $x$ . Applying Lemma 3 on the eq. (16), we obtain

$$(16) \geq \mathbb{E}\{h'_1(\sigma(A_1))\} \mathbb{E}\{h'_2(\sigma(A_2))\}$$

for increasing functions  $h_1$  and  $h_2$ , and the same inequality also holds for both decreasing functions. This completes the main statement of this Lemma.  $\square$

Recursively applying the above relation, one can easily derive the following lemma, which is a generalized version of Lemma 4 to the case of multiple disjoint subsets. We omit the detailed proof as it is trivial to verify.

**Lemma 5.** *For any  $-\infty < k_0 < k_1 < \dots < k_T < \infty$ , define disjoint subsets  $A_i = [k_i, \dots, k_{i+1} - 1]$  for  $i = 0, \dots, T-1$ . For all increasing (or all decreasing) functions  $h_i$ ,  $i \in [0, \dots, T-1]$  each taking arguments from  $\sigma(A_i) \triangleq \{\sigma(k_i), \dots, \sigma(k_{i+1} - 1)\}$ , we have*

$$\mathbb{E}\left\{\prod_{i=0}^{T-1} h_i(\sigma(A_i))\right\} \geq \prod_{i=0}^{T-1} \mathbb{E}\{h_i(\sigma(A_i))\}.$$

Let  $B_i \triangleq \{-t+i, -t+i+T, \dots, -t+i+j_i^*T : -t+i+j_i^*T \leq 0\}$  for  $i \in [0, 1, \dots, T-1]$  be the index sets of time slots in the range  $[-t, 0]$  each has the same modulo  $T$ . Clearly,  $\cup_{i=0}^{T-1} B_i \equiv [-t, 0]$ . Define  $B'_0 = [-t, -t+|B_0|-1]$ , and  $B'_i = [-t+\sum_{0 \leq k \leq i-1} |B_k|, -t+\sum_{0 \leq k \leq i} |B_k|-1]$  for  $i = 1, \dots, T-1$ . We derive the following

$$\begin{aligned} \mathbb{E}[e^{-\theta S^{\text{std}}(t)}] &= \mathbb{E}[e^{-\theta \sum_{i=-t}^0 \sigma(i)}] = \mathbb{E}[e^{-\theta \sum_{i=0}^{T-1} \sum_{j \in B'_i} \sigma(j)}] \\ &= \mathbb{E}\left[\prod_{i=0}^{T-1} e^{-\theta \sum_{j \in B'_i} \sigma(j)}\right] \stackrel{(a)}{\geq} \prod_{i=0}^{T-1} \mathbb{E}[e^{-\theta \sum_{j \in B'_i} \sigma(j)}] \\ &\stackrel{(b)}{=} \prod_{i=0}^{T-1} \mathbb{E}[e^{-\theta \sum_{j \in B_i} \sigma^{(T)}(j)}] = \mathbb{E}[e^{-\theta S^{\text{del}}(t)}] \end{aligned}$$

where the (a) is due to that  $\sigma(\cdot) = \mathbf{1}\{\sigma(\cdot) \in B_v\}$  is an increasing function and thus  $e^{-\theta \sum_{j \in B_i} \sigma(j)}$  is decreasing from which the inequality follows from Lemma 5 with our choices  $h_i(\sigma(A_i)) = e^{-\theta \sum_{j \in A_i} \sigma(j)}$  and  $A_i = B'_i$ , and (b) follows since in the steady state, the distribution of schedules at  $i$ -th Markov chain up to time  $l$ ,  $\sigma^{(T)}(i), \sigma^{(T)}(i+T), \dots, \sigma^{(T)}(i+lT)$ , for  $i \in 0, \dots, T-1$  is equivalent to that of schedule by the ordinary Markov chain with  $T = 1$ ,  $\sigma(0), \sigma(1), \dots, \sigma(l)$ .

**Proof of  $\mathbb{E}[e^{-\theta S^{\text{del}}(t)}] \geq \mathbb{E}[e^{-\theta S^{\text{AC}}(t)}]$ :** We denote by  $\hat{\sigma}^{(T)}(s)$  the schedule at time  $s$  determined by the AC-CSMA algorithm with the parameter  $T$ . Let  $i_0$  be the time index that the two algorithms starts, and let  $x_{i_0}, \dots, x_{i_0+T-1}$  be the first  $T$  initial schedule states where  $x_i \in \Omega$  for  $i = i_0, \dots, i_0 +$

$T-1$ . Define by  $C_i \triangleq \{i_0+i, i_0+i+T, \dots, i_0+i+\hat{j}_i T : i_0+i+\hat{j}_i T \leq 0\}$  the index sets of time slots each has the same modulo  $T$  in the range of  $[i_0, 0]$ .

**Lemma 6.** *For all increasing (or all decreasing) functions  $h_i$ , we have*

$$\mathbb{E}\left\{\prod_{i=0}^{T-1} h_i(\hat{\sigma}^{(T)}(C_i))\right\} \leq \prod_{i=0}^{T-1} \mathbb{E}\{h_i(\sigma^{(T)}(C_i))\} \quad (17)$$

where  $h_i : \Omega^{|C_i|} \rightarrow \mathbb{R}$ ,  $i = 0, \dots, T-1$  is increasing if  $h_i(\sigma(C_i)) \leq h_i(\sigma'(C_i))$  (decreasing if  $h_i(\sigma(C_i)) \geq h_i(\sigma'(C_i))$ ) whenever  $\sigma(j) \preceq_{\Omega} \sigma'(j)$  for all  $j \in C_i$ .

*Proof.* Denote by  $U_i^{\text{NA}}(t)$  negatively associated uniform random numbers generated by node  $i$  at time  $t$ . For the state update function of delayed CSMA algorithm and AC-CSMA with the same parameter  $T$ , we can write  $\sigma^{(T)}(t) = F(\sigma^{(T)}(t-T), \mathbf{m}(t), \vec{U}(t))$  and  $\hat{\sigma}^{(T)}(t) = F(\hat{\sigma}^{(T)}(t-T), \mathbf{m}(t), \vec{U}^{\text{NA}}(t))$ , respectively.

Consider an event  $E$  in which the initial  $T$  schedule states are given and all the decision schedules up to time  $t$  are realized such that  $\{\{\hat{\sigma}^{(T)}(s) = x_s\}_{i_0 \leq s \leq i_0+T-1}, \{\mathbf{m}(s) = m_s\}_{i_0 \leq s \leq t}\} \triangleq E$ . Under the event  $E$ ,  $\hat{\sigma}^{(T)}(s+T) = F_{[m_{s+T}]}(x, \vec{U}^{\text{NA}}(s+T))$  is increasing in  $\vec{U}^{\text{NA}}(s+T)$  for  $i_0 \leq s \leq t$  in the sense that  $F_{[m_{s+T}]}(x, \vec{U}^{\text{NA}}(s+T)) \preceq_{\Omega} F_{[m_{s+T}]}(x, \vec{U}'^{\text{NA}}(s+T))$  when  $\vec{U}^{\text{NA}}(s+T) \preceq_{\mathbb{R}} \vec{U}'^{\text{NA}}(s+T)$ ;  $F_{[m_{s+2T}]}(F_{[m_{s+T}]}(x, \vec{U}^{\text{NA}}(s+T)), \vec{U}^{\text{NA}}(s+2T)) \preceq_{\Omega} F_{[m_{s+2T}]}(F_{[m_{s+T}]}(x, \vec{U}'^{\text{NA}}(s+T)), \vec{U}'^{\text{NA}}(s+2T))$  when  $\vec{U}^{\text{NA}}(s+2T) \preceq_{\mathbb{R}} \vec{U}'^{\text{NA}}(s+2T)$ ; and so on. Hence, under the condition  $E$ ,  $h_i(\hat{\sigma}^{(T)}(C_i))$  can be represented by an another increasing function  $h'_i(\vec{U}^{\text{NA}}(s), s \in C_i)$ . Define  $\mathcal{F} \triangleq \{\{\hat{\sigma}^{(T)}(s)\}_{i_0 \leq s \leq i_0+T-1}, \{\mathbf{m}(s)\}_{i_0 \leq s \leq t}\}$ , and observe the following,

$$\begin{aligned} \mathbb{E}\left\{\prod_{i=0}^{T-1} h_i(\hat{\sigma}^{(T)}(C_i))\right\} &= \mathbb{E}\left\{\mathbb{E}\left\{\prod_{i=0}^{T-1} h_i(\hat{\sigma}^{(T)}(C_i)) | \mathcal{F}\right\}\right\} \\ &= \mathbb{E}\left\{\mathbb{E}\left\{\prod_{i=0}^{T-1} h'_i(\vec{U}^{\text{NA}}(s), s \in C_i) | \mathcal{F}\right\}\right\} \\ &\stackrel{(a)}{\leq} \mathbb{E}\left\{\prod_{i=0}^{T-1} \mathbb{E}\{h'_i(\vec{U}^{\text{NA}}(s), s \in C_i) | \mathcal{F}\}\right\} \\ &\stackrel{(b)}{=} \prod_{i=0}^{T-1} \mathbb{E}\left\{\mathbb{E}\{h'_i(\vec{U}^{\text{NA}}(s), s \in C_i) | \mathcal{F}\}\right\} \\ &= \prod_{i=0}^{T-1} \mathbb{E}\{h_i(\hat{\sigma}^{(T)}(C_i))\} \stackrel{(c)}{=} \prod_{i=0}^{T-1} \mathbb{E}\{h_i(\sigma^{(T)}(C_i))\} \end{aligned}$$

where (a) is the Property ( $P_2$ ) of NA in [32] which is simply repeated application of Definition 1, (b) is due to the fact that the distribution of decision schedule for each time slot is i.i.d. and that of each of initial  $T$  schedule state is given by  $x_{i_0}, \dots, x_{i_0+T-1}$  each with probability 1, and (c) follows because the time index sets  $\{C_i\}_{0 \leq i \leq T-1}$  do not share any common time slot, i.e.,  $C_i \cap C_j = \emptyset$ , for  $i \neq j$ , and

the marginal distribution of  $\hat{\sigma}^{(T)}(C_i)$  is identical to that of  $\sigma^{(T)}(C_i)$ , for all  $i = 0, \dots, T-1$ .  $\square$

Note that Lemma 6 holds for any given initial time  $i_0$ , and hence the result also holds in the steady state. Therefore, we have

$$\begin{aligned} \mathbb{E}[e^{-\theta S^{\text{AC}}(t)}] &= \mathbb{E}\left[\prod_{i=0}^{T-1} e^{-\theta \sum_{j \in B_i} \hat{\sigma}^{(T)}(j)}\right] \\ &\stackrel{(a)}{\leq} \prod_{i=0}^{T-1} \mathbb{E}[e^{-\theta \sum_{j \in B_i} \sigma^{(T)}(j)}] = \mathbb{E}[e^{-\theta S^{\text{det}}(t)}] \end{aligned}$$

where (a) is due to Lemma 6 with the choice of  $h_i(\hat{\sigma}^{(T)}(C_i)) = e^{-\theta(\sum_{j \in B_i} \hat{\sigma}(j) + \sum_{j \in C_i \setminus B_i} 0)}$  which is decreasing function according to the definition in the Lemma.  $\square$