Self-Synchronizing Properties of CSMA Wireless Multi-hop Networks

Kuang Xu
MIT, LIDS
Cambridge, MA 02139
kuangxu@mit.edu

Olivier Dousse
Nokia Research Center
Lausanne, Switzerland
olivier.dousse@nokia.com

Patrick Thiran
EPFL
Lausanne, Switzerland
patrick.thiran@epfl.ch

ABSTRACT

We show that CSMA is able to spontaneously synchronize transmissions in a wireless network with constant-size packets, and that this property can be used to devise efficient synchronized CSMA scheduling mechanisms *without* message passing. Using tools from queuing theory, we prove that for any connected wireless networks with arbitrary interference constraints, it is possible to implement self-synchronizing TDMA schedules *without any* explicit message passing or clock synchronization besides transmitting the original data packets, and the interaction can be fully local in that each node decides when to transmit next only by overhearing its neighbors' transmissions. We also provide a *necessary and sufficient condition* on the emergence of self-synchronization for a given TDMA schedule, and prove that such conditions for self-synchronization can be checked in a finite number of steps for a finite network topology.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless Communication, Distributed Networks, Network Topology

General Terms

Design, Performance, Algorithms

Keywords

Stochastic recursive sequence, Self-synchronization, Scheduling algorithm

1. INTRODUCTION

Wireless multi-hop mesh networks have received a lot of attention over the last years in the research community. In particular, the problem of scheduling transmissions efficiently given certain interference constraints have been addressed in several papers and algorithms to solve this problem have been proposed [13, 21, 3, 19, 1, 17, 4]. However,

in most cases (in particular in the papers cited above), the wireless channel is assumed to be *slotted*, which requires in practice that the wireless nodes are somehow *synchronized*. The realization of this synchronization requires some communication between nodes and thus induces some overhead.

In this paper, we show that the information acquired by nodes through a standard carrier sensing mechanism is enough to guarantee a robust synchronization and allows to execute transmission schedules without any explicit synchronization message. More precisely, we show that under certain conditions, if nodes are aware of their neighbors and of the sequence in which those are supposed to transmit, then a raw CSMA protocol is enough to maintain a synchronized slotted execution of the prescribed schedule. As a consequence, no explicit synchronization messages are required. We also detail the conditions that the contention graph and the schedule must fulfill for this implicit synchronization to occur. In that sense, our work complements the research on scheduling in mesh networks.

The self-synchronization property shown in this paper is well illustrated in the following example. Consider the 6-node-ring network depicted in Figure 1. The vertices represent wireless nodes implementing CSMA, and the edges represent interference constraints: that is, if the node at one end of an edge emits, the node at the other end will sense the channel to be busy and refrain from emitting[1]. As in many CSMA systems, we let nodes wait for a random (but small) amount of time after they sense the channel to be idle before emitting; if the channel is still idle after this random waiting time, they actually start their transmission.

We assume that all 6 nodes have an infinite backlog of equally sized packets to send. However, in order to enforce some fairness in the network, we add the following simple rule on top of the normal CSMA operation: after emitting, a node waits until it detects the end of a transmission from a neighboring node before it tries to emit again.

We simulate such a network in continuous time. The key observation is that a highly synchronized transmission pattern quickly emerges, as one can see on Figure 2. Note that the sets of nodes $\{1, 3, 5\}$ and $\{2, 4, 6\}$ form an alternating sequence of concurrent transmissions. Strikingly, it appears *as if* a time-division-multiple-access (TDMA) schedule is enforced: the set of nodes $\{1, 3, 5\}$ are "told" to transmit during one "time slot", and nodes $\{2, 4, 6\}$ during the next. Also, all concurrent transmissions in either set seem to start

---

[1]We start with this simplistic example for the sake of clarity, although it does not necessarily capture all aspects of a real wireless network.
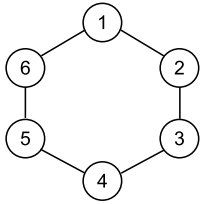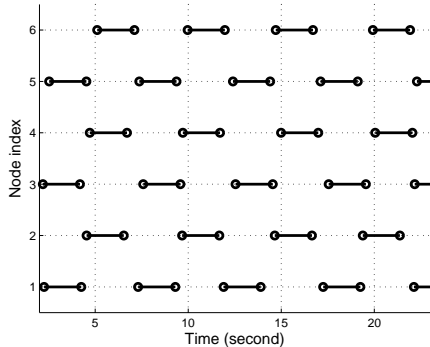
Figure 1: A 6-node-ring topology.

Figure 2: The continuous time transmission pattern. A circle indicates the starting or finish of a transmission.

almost *simultaneously*. However, such TDMA mechanism is nowhere specified in the protocol. The protocol is fully local in the sense that each node makes transmission decisions based solely on the activities of its immediate neighbors. In particular, *no explicit clock synchronization* is implemented, contrary to the case of a standard TDMA scheme.

Why do such local iterations lead to a "global" synchronization of transmission patterns? More importantly, can we implement a distributed CSMA protocol that automatically guarantees a stable synchronized transmission pattern to achieve *zero collision*, but without any of TDMA's expensive explicit synchronization? We will prove positive answers to this design challenge.

In this paper, we consider a setting close to the classical spatial TDMA case where nodes in the network follow a shared deterministic schedule, $\mathcal{L}$, which consists of a periodic sequence of subsets of the set of all nodes and is distributed to each node at beginning of time. However, instead of looking at its own clock to decide when to transmit next, like in a slotted TDMA, a node will simply transmit after *sensing* all of its neighbors have completed their assigned transmissions, according to the schedule. What conditions, then, does $\mathcal{L}$ have to satisfy in order for all concurrent transmissions to stay synchronized? Furthermore, for an arbitrary interference topology, is it always possible to come up with such a schedule $\mathcal{L}$ so that the locally enforced schedule can guarantee global synchronization? Our results in the next sections will answer both questions. The major contributions of this paper are summarized as follows:

- We introduce tools from stochastic recursive sequences and precedence graphs to analyze the self-synchronization phenomenon in CSMA (Section 4). We prove a necessary and sufficient condition for a scheduling frame

to be "synchronizable" by a locally enforced CSMA protocol (Section 5). Given a finite contention graph, we show that this criterion of synchronization can be checked in a finite number of steps. In particular, this means that it is possible to implement a given TDMA scheduling frame without any explicit clock synchronization.

- We show that for *any arbitrary* connected interference topology, there exists a scheduling frame that achieves global self-synchronization by using a locally enforced protocol (Section 6). The proof we provide is *constructive* and can be used as a first step towards algorithms in generating self-synchronizing schedule frames for CSMA wireless networks. We also illustrate our results by applying them to two concrete examples (Section 7).

2. RELATED WORK

Synchronization in channel access has been widely acknowledged to be an important factor of optimality in the utilization of the wireless spectrum. Typically, slotted protocols achieve better throughput than their random counterparts, the most famous example being Aloha and Slotted Aloha [18]. Therefore, a large part of the literature on wireless multi-hop protocols is based on the slotted paradigm.

Already in 1985, Nelson and Kleinrock [13] created the first spatial TDMA scheme for wireless multi-hop networks. Such scheduling schemes have recently received much attention in the context of mesh networking (see for example [21, 3, 1, 19, 17, 4]). However, in these papers, synchronization needs to be achieved through some separate dedicated mechanism. To mitigate this overhead, hybrid CSMA/TDMA protocols have been proposed in [16] where explicit synchronization is still required, but with looser accuracy constraints. The concept of "slotted CSMA" appeared in the IEEE 802.15.4 standard [5, 14], but is also using explicit synchronization of nodes, in order to save energy in this case.

Several methods have been explored to synchronize nodes' clocks. The most common are explicit time exchange protocols, such as NTP, which are reviewed in [8]. However, these methods generate large traffic overhead. In order to design light-weight synchronization mechanisms, researchers have looked at methods inspired by fireflies: In 1990 Mirollo and Strogatz modeled fireflies with pulse-coupled oscillators (PCO) [11]. Such oscillators have later been used to synchronize the nodes of wireless networks [9, 12].

Very few attempts have been made to analyze or exploit the self-synchronization properties of CSMA. In [20], Singh et al. exploit the periodic nature of VoIP traffic to create some implicit synchronization of CSMA nodes. In [15], Raman and Chebrolu use a similar synchronization property of CSMA as in the present paper in a two-phase schedule. In [6], self-synchronization of CSMA networks is observed in linear networks by simulation only. Our work essentially generalizes the intuition of these two papers.

The process through which CSMA nodes synchronize is actually similar to that of some queuing networks. Therefore, we make use of concepts of queuing theory to solve our problem. In particular, Baccelli and Liu develop a theory in [2], which we will use extensively in Section 4 to analyze CSMA networks.

3. MODEL & CONCEPTS

In this section, we describe more formally the model and assumptions presented in the introductory example. Then we introduce the basic concepts that are needed to prove our results.

3.1 Network Model

We represent the wireless network with its *contention graph* $\mathcal{G}_I = (\mathcal{E}_I, \mathcal{V}_I)$. In this graph, vertices $\mathcal{V}_I$ represent a transmission entity, and edges $(i,j), i, j \in \mathcal{V}_I$ represent the exclusion rules between these entities necessary to prevent collisions. Transmission entities can either be the *physical wireless devices* themselves, as in the motivating example (this case corresponds to the *primary interference model* used in [10], also called *node-exclusive spectrum sharing*), or they can be *links* between physical devices (*secondary interference model* or *link-exclusive spectrum sharing*). The second case is particularly suitable to model 802.11 networks, where the RTS-CTS handshake mechanism allows to prevent neighbors or both the emitter and receiver to refrain from transmitting (see e.g. [7] for such a modeling of 802.11 and [21] for a general discussion about the two cases).

Irrespective of the true nature of the transmission entities (called hereafter "nodes"), we will model the CSMA system as follows: We say that a node $i \in \mathcal{V}_I$ is "activated" (or simply "i transmits") when a transmission is ongoing at i. The carrier sensing mechanism ensures that when i transmits, all neighbors of i in $\mathcal{G}_I$ (we denote this set by $\mathcal{N}_I(i)$) are prevented from transmitting. Furthermore, we assume that a node is able to *detect* the starting of transmissions of *all* of its neighbors in $\mathcal{G}_I$. This is reasonable due to the interference. However, some additional signal processing (e.g. power envelope analysis) may be required to detect the starting of transmissions when multiple neighbors are transmitting at the same time. Finally, we assume that each node has an infinite backlog of packets, so that they always transmit a packet when they are scheduled to do so. Alternatively, nodes can be assumed to send dummy packets whenever their turn to emit comes and their queue is empty; they may do so just to keep the whole network synchronized[2].

In order to observe a synchronization effect, we impose that *constant-size* packets are being transmitted, each taking D seconds to be sent[3].

3.2 MIS Schedules

Consider the contention graph $\mathcal{G}_I = (\mathcal{E}_I, \mathcal{V}_I)$ introduced in Subsection 3.1. Let

$$L = (L_1, L_2, \ldots, L_K)$$

be a **scheduling frame**, where each L_i, called a **slot of concurrent transmissions**, is a subset of $\mathcal{V}_I$ for all $1 \leq i \leq K$. The frame is then repeated to generate the **schedule**

$$\mathcal{L} = (L_1, L_2, \ldots, L_K, L_1, L_2, \ldots).$$

[2]This of course may not be energy-efficient if the traffic is extremely low, but is reasonable when there is a continuous flow of traffic.

[3]We will see later in Subsection 3.3 that the transmission time does not need to be exactly equal to D, but simply that the transmissions last for D plus some bounded random variable that is small compared to D.

We denote the ith set in the schedule by $\mathcal{L}(i)$. Note that as a result of repetition

$$\mathcal{L}(i) := L_{i \bmod (K+1)}, \forall i \in \mathbb{N}.$$

Observe that due to the interference constraints imposed by the contention graph $\mathcal{G}_I$, each slot of concurrent transmissions, $\mathcal{L}(t)$, must be an *independent set* of $\mathcal{G}_I$, in that no two elements in $\mathcal{L}(t)$ reside on the ends of the same edge in $\mathcal{G}_I$. This constraint has been common in the classic wireless scheduling literature. In fact, it is often the case that the set of concurrent transmission nodes are configured to be a **maximal independent set (MIS)** of the contention graph in order to achieve maximal spatial re-use, where a maximal independent set is an independent set that is not a subset of any other independent set.

In our case, using slots (L_i) that are maximal independent sets of $\mathcal{G}_I$ has the additional benefit which allows a schedule $\mathcal{L}$ to be *enforced locally*: if L_i is a maximal independent set of $\mathcal{G}_I$ for all $1 \leq i \leq K$, then a node $n \in L_i$ overhears (or equivalently, interferes with) at least one node in each of the $L_j, j \neq i$. As a consequence, in the simple CSMA protocol to be introduced shortly, a node will be able to decide when to transmit next solely by on overhearing its neighbors' transmissions. Therefore, from this point on we focus on the class of **maximal-independent-set (MIS) schedules**, where each slot $\mathcal{L}(t)$ forms a maximal independent set of the contention graph $\mathcal{G}_I$, defined as follows:

Definition 1. *A schedule* $\mathcal{L} = (L_1, L_2, ..., L_K, L_1, ...)$ *is an* **MIS schedule** *if the scheduling frame* $L = (L_1, \ldots, L_K)$ *satisfies:*

●**(Fairness)** $\bigcup_{1 \leq i \leq K} L_i = \mathcal{V}_I$,

●**(MIS)** L_i *is a maximal independent set (MIS) of* $\mathcal{G}_I$, *for all* $1 \leq i \leq K$. *That is, each node in the network is scheduled to transmit at least once per frame, and each slot corresponds to a maximal independent set of the contention graph.*

3.3 Local Enforcement of MIS Schedules

Suppose every node is *given* the MIS schedule $\mathcal{L}$. Consider the following simple continuous-time protocol:

A Locally Enforced MIS Schedule

For all nodes $v \in \mathcal{V}_I$,

1. If $v \in \mathcal{L}(1)$, it starts transmitting at some time $S(v,1)$, given that $\max_{u,v \in \mathcal{L}(1)} |S(v,1) - S(u,1)| < \gamma \ll D$.

2. If $v \in \mathcal{L}(t), t \geq 2$, it keeps listening to the channel until all of its neighbors in $\mathcal{L}(t-1)$ have completed their *assigned* transmissions. Node v then starts its own transmission after a bounded **random delay** $C(v,t)$. Each $C(v,t)$ is independently distributed on $\mathbb{R}^+$ and $C(v,t) \leq \delta, \forall t, v$. We label the **starting time** of this transmission initiated by v in $\mathcal{L}(t)$ as $S(v,t)$.

Notice that while random backoffs are often used in traditional contention-based CSMA schemes to avoid collisions, our protocol has no explicit need for random backoffs to be implemented. Indeed, each slot $\mathcal{L}(t)$ is an independent set of the contention graph, and is hence by definition collision-free. In this context, the random delay $C(v,t)$ can be interpreted as the aggregation of all delays caused by physical imperfections such as sensing delays, small deviations of packet transmission time from the theoretical value D, etc. Of course, it could also be the case that the system designer prefers to keep the random backoff mechanism as an option.

Requirement (1) in the above protocol basically states that the initial state is roughly synchronized: the very first round of transmissions all start within a time interval of size γ from each other. Note that (2) is possible because all slots are MIS: for every node $v \in \mathcal{L}(t+1)$, there is at least one neighbor of v that is scheduled in $\mathcal{L}(t)$, for otherwise v can be added to $\mathcal{L}(t)$ without causing any interference, contradicting with $\mathcal{L}(t)$ being a maximal independent set of $\mathcal{G}_I$.

Up until this point, the feasibility of this local protocol has not been justified. The actual sequence of concurrent transmissions may not be the same as $\mathcal{L}$, in which case the labeling of starting times and random delays are not well defined. As we will see at the end of Subsection 3.5, this issue can be resolved when a *synchronized schedule* is enforced.

3.4 Precedence Graph Representation of Concurrent Transmissions

Definition 2. *A* **precedence graph** *is a directed graph* $\mathcal{G}_P = (\mathcal{V}_P, \mathcal{E}_P)$ *where:*

- *The set of vertices $\mathcal{V}_P$ is partitioned into disjoint "levels" $\{V_P(t)\}_{t \in \mathbb{N}}$, such that*

$$\mathcal{V}_P = \bigcup_{t \in \mathbb{N}} V_P(t),$$

$$V_P(t_1) \cap V_P(t_2) = \emptyset, \forall t_1 \neq t_2.$$

- *Denote by $(\mathbf{v}_1, \mathbf{v}_2)$ a directed edge from vertices $\mathbf{v}_1$ to $\mathbf{v}_2$. Then for all $t \in \mathbb{N}$,*

$$(\mathbf{v}_1, \mathbf{v}_2) \in \mathcal{E}_P \text{ only if } \mathbf{v}_1 \in V_P(t), \mathbf{v}_2 \in V_P(t+1).$$

For our purpose, assume that each level of the precedence graph, $V_P(t)$, is non-empty with a cardinality $M(t) \geq 1$ for all $t \in \mathbb{N}$, and hence can be indexed by the set $\mathcal{M}_P(t) := \{1, 2, ..., M(t)\}$. We label a vertex $\mathbf{v} \in V_P(t)$ by the pair (v, t) where $v \in \mathcal{M}_P(t)$ is the vertex's corresponding index at level t. See Figure 4 for an example of a precedence graph.

We now make the connection between the MIS schedule $\mathcal{L}$ and the precedence graph introduced above. We would like each level of the precedence graph $V_P(t)$ to represent the t-th slot of *concurrent transmissions*, $\mathcal{L}(t)$. We would also want the edges between $V_P(t)$ and $V_P(t+1)$ to capture the interference constraints. Let v be a node of the contention graph $\mathcal{G}_I$. Suppose $v \in \mathcal{L}(t+1)$, i.e. node v is scheduled to transmit at the $(t+1)$-th slot. Then node v *cannot* start its transmission until *all* of its neighbors who were scheduled to transmit in the previous slot $\mathcal{L}(t)$, i.e.

$$\mathcal{L}(t) \cap \mathcal{N}_I(v)$$

have completed their assigned transmissions. As we will soon observe, this interference constraint can be enforced by adding directed edges in the precedence graph that go from the vertices that correspond to the transmissions of $\mathcal{L}(t) \cap \mathcal{N}_I(v)$, to the vertex $(v, t+1)$. We formalize this intuition by the following construction of a precedence graph $\mathcal{G}_P$ based on the MIS schedule $\mathcal{L}$.

Precedence Graph Induced by MIS Schedule

1. For all $t \in \mathbb{N}$, let $V_P(t)$ have the same cardinality as $\mathcal{L}(t)$, and label $V_P(t)$ accordingly as:

$$V_P(t) = \{(v, t) : v \in \mathcal{L}(t)\}, \forall t \in \mathbb{N}.$$

2. Assign a directed edge from $(v_t, t) \in V_P(t)$ to $(v_{t+1}, t+1) \in V_P(t+1)$ if $v_t = v_{t+1}$, or v_t and v_{t+1} share an edge in the contention graph, i.e. $v_t \in \mathcal{N}_I(v_{t+1})$.

3. Assign to each vertex in the precedence graph, (v, t), the corresponding starting time, $S(v, t)$, as defined in the locally enforced protocol in Section 3.3.

If a precedence graph $\mathcal{G}_P$ is constructed by the above procedure, we say that $\mathcal{G}_P$ is **induced** by the MIS schedule $\mathcal{L}$ and contention graph $\mathcal{G}_I$. See Figure 4 and Figure 5 for examples of precedence graphs induced by MIS schedules.

We now arrive at the following important recursive relations that connect the sequence of starting times with the sequence of random delays. Based on the way $\mathcal{L}$ is enforced, it is easy to verify that the sequence of starting times $\{S(v, t)\}_{v \in \mathcal{L}(t), t \in \mathbb{N}}$ satisfies:

For all $v \in \mathcal{L}(t), t \in \mathbb{N}, t \geq 2$,

$$S(v, t) = \max_{u \in \mathcal{L}(t-1), (u,v) \in \mathcal{E}_P} \{S(u, t-1)\} + D + C(v, t), \quad (1)$$

where $\{C(v, t)\}_{t \in \mathbb{N}, v \in \mathcal{L}(t)}$ is the sequence of random delays, $S(v, 1)$ is the set of initially synchronized starting times, and the constant D is the amount of time it takes for a packet to be transmitted. Under the maximization of Equation (1), (u, v) is used as a shorthand for the directed edge from $(u, t-1)$ to (v, t) in $\mathcal{E}_P$.

Notations: The notations below will be followed throughout the paper:

- A **node** always refers to a vertex on the contention graph $\mathcal{G}_I$, which represents a physical node or entity. A **vertex** alway refers to a vertex $\mathbf{v} = (v, t)$ on the precedence graph $\mathcal{G}_P$, which represents a particular transmission.

- When a precedence graph $\mathcal{G}_P$ is induced by some MIS schedule $\mathcal{L}$, since each level $V_P(t)$ of $\mathcal{G}_I$ is indexed by slot $\mathcal{L}(t)$, the following are used interchangeably:

$$v \in \mathcal{L}(t) \Leftrightarrow (v, t) \in V_P(t).$$

In general, we use $\mathcal{L}(t)$ when focusing on the MIS schedule, and $V_P(t)$ when speaking of properties on the precedence graph.

3.5 Definition of Synchronized MIS Schedules

Finally, we formally define what is it meant to have synchronization when an MIS schedule $\mathcal{L}$ is locally enforced as described in the Subsection 3.2.

Definition 3. *An MIS schedule $\mathcal{L}$ for a finite contention graph $\mathcal{G}_I$ is said to be* **synchronized** *when executed with the local protocol, if and only if the following holds: Given the precedence graph $\mathcal{G}_P$ induced by $\mathcal{L}$ and $\mathcal{G}_I$, for all $\delta > 0, \gamma > 0$, there exists $\beta(\delta, \gamma) > 0, \epsilon(\delta) > 0$ and $T(\mathcal{G}_P) \in \mathbb{N}$ such that:*

(Transient State) *For all $1 \leq t < T(\mathcal{G}_P)$*

$$\max_{u, v \in \mathcal{L}(t)} |S(u, t) - S(v, t)| \leq \beta(\delta, \gamma) \ a.s. \quad (2)$$

(Steady State) *For all $t > T(\mathcal{G}_P)$*

$$\max_{u, v \in \mathcal{L}(t)} |S(u, t) - S(v, t)| \leq \epsilon(\delta) \ a.s. \quad (3)$$

whenever $\max_{u,v \in \mathcal{L}(1)} |S(u,1) - S(v,1)| \leq \gamma$ *a.s., and the random delays* $|C(v,t)| \leq \delta$ *a.s.* $\forall t \in \mathbb{N}, v \in \mathcal{L}(t)$. *In addition,*

$$\lim_{\delta,\gamma \to 0} \beta(\delta,\gamma) = 0, \lim_{\delta \to 0} \epsilon(\delta) = 0. \qquad (4)$$

This definition of synchronization basically specifies the following: There exists a finite time $T(\mathcal{G}_P)$, a function only of the precedence graph induced by the MIS schedule $\mathcal{L}$, which divides time into the transient and steady states. In the transient state ($t < T(\mathcal{G}_P)$), the maximum difference in starting times among nodes in $\mathcal{L}(t)$ is uniformly bounded by a function of both the initial condition γ, and the maximum individual random delay, δ. In fact, this follows trivially from the fact that $T(\mathcal{G}_P)$ is finite and all random delays are bounded by δ. However, in the steady state ($t \geq T(\mathcal{G}_P)$), there exists a uniform bound for the maximum difference in starting times that is a function of *only* δ, which diminishes to zero as δ goes to zero. It is easy to see that latter bound for the steady state is much stronger, and it will be our main focus in the rest of the paper.

In general, this definition requires that in a synchronized schedule the differences in starting times in the steady state can be bounded arbitrarily small by having small bounded random delay distributions. Conversely, if a schedule is not synchronized, a sequence of bounded random delays can lead to arbitrarily large differences in starting times with positive probability as t increases.

Roughly speaking, when a *synchronized schedule* is locally enforced, the random delays occurring at each stage will not "accumulate" as time goes on, to eventually drive apart the starting times of transmissions in some future slot. Observe that any MIS schedule is trivially synchronized in the ideal case where the random delays $C(v,t) = 0, \forall t \geq 2, v \in \mathcal{L}(t)$; the protocol basically reduces to a (trivially) slotted TDMA. This is obviously not the case in reality.

Can one always find an MIS schedule that achieves synchronization with the presence of positive random delay distributions on an arbitrary contention graph? If so, what structure must the schedule $\mathcal{L}$ possess with respect to the underlying contention topology $\mathcal{G}_I$? To obtain concrete answers, we use tools from the theory of stochastic recursive sequences to study the behavior of the starting times $\{S(v,t)\}_{t \in \mathbb{N}}$, which will be introduced in the next section.

Finally, the feasibility of locally enforced protocol, as was described in Subsection 3.3, can be justified if $\mathcal{L}$ is synchronized by the above definition: We simply need to make sure that the (constant) transmission time for a packet (D) satisfies

$$D > \max\{\beta(\delta,\gamma), \epsilon(\delta)\}.$$

In this case, the sequence of concurrent transmissions will indeed be the same as $\mathcal{L}(t)$ a.s., and the assignments of $\{S(v,t)\}_{t \in \mathbb{N}}$ and $\{C(v,t)\}_{t \in \mathbb{N}}$ are well defined. Indeed, the quantities γ, δ are often constrained by nature in reality, while the system designer has control over the packet length D.

4. STOCHASTIC RECURSIVE SEQUENCES ON PRECEDENCE GRAPHS

The ultimate goal of this section (Theorem 7) is to develop an upper-bound on the *maximum difference in starting times*, $\max_{v_1,v_2 \in \mathcal{L}(t)} |S(v_1,t) - S(v_2,t)|$, together with a sufficient condition (total connectedness) on the precedence graph for the upper-bound to hold. The theory of stochastic recursive sequences [2], which has been used in modeling task completions in queuing networks and parallel computing, will serve as a powerful tool in our analysis. We begin by studying some properties of the precedence graph $\mathcal{G}_P$.

It is easy to see that $\mathcal{G}_P$ is an acyclic graph, in which the notion of a (directed) path between two vertices is well defined . Let $\Gamma_{(v_1,t_1)}^{(v_2,t_2)}$ be the **set of all paths** from (v_1,t_1) to (v_2,t_2), $t_1 < t_2$:

$$\Gamma_{(v_1,t_1)}^{(v_2,t_2)} = \{(\mathbf{v}_{t_1}, \mathbf{v}_{t_1+1}, ..., \mathbf{v}_{t_2}) : \mathbf{v}_i \in \mathcal{V}_P(i), \forall t_1 \leq i \leq t_2,$$
$$(\mathbf{v}_i, \mathbf{v}_{i+1}) \in \mathcal{E}_P, \forall t_1 \leq i \leq t_2 - 1, \mathbf{v}_1 = (v_1, t_1),$$
$$\mathbf{v}_{t_2} = (v_2, t_2)\}.$$

Notice that from the definition of a precedence graph (Definition 2), any path between (v_1,t_1) and (v_2,t_2), if exists, has the **same length** which is equal to $t_2 - t_1$.

The following notion of descendant and ancestor sets on a precedence graph $\mathcal{G}_P = (\mathcal{V}_P, \mathcal{E}_P)$ will be used repeatedly throughout:

Definition 4. *For* $t_1 < t_2$, *the* **descendant set** *of* (v_1,t_1) *at level* t_2, *denoted by* $d_{(v_1,t_1)}(t_2)$, *is the subset of* $\mathcal{L}(t_2)$ *to which a path from* (v_1,t_1) *exists* [4],

$$d_{(v_1,t_2)}(t_2) = \left\{v \in \mathcal{L}(t_2) : \Gamma_{(v_1,t_1)}^{(v,t_2)} \neq \emptyset\right\}.$$

Conversely, for $t_1 < t_2$, *the* **ancestor set** *of a vertex* (v_2,t_2) *at level* t_1, *denoted by* $a_{(v_2,t_2)}(t_1)$, *is the subset of* $\mathcal{L}(t_1)$ *form which a path to* (v_2,t_2) *exists,*

$$a_{(v_2,t_2)}(t_1) = \left\{v \in \mathcal{L}(t_1) : \Gamma_{(v,t_1)}^{(v_2,t_2)} \neq \emptyset\right\}.$$

Lemma 5 is essentially a variation of Lemma 2.3 in [2].

Lemma 5 (Decomposition of Stochastic Recursive Sequences). *Assuming that every* $(v,t) \in \mathcal{V}_P(t), t \geq 2$ *has at least one incoming edge from some vertices in* $\mathcal{V}_P(t-1)$, *i.e.* $a_{(v,t)}(t-1) \neq \emptyset$, *the following holds:*

$$S(v,t) = \max_{w \in a_{(v,t)}(t-k)} \left\{S(w, t-k) + L_{(w,t-k)}^{(v,t)}\right\} + kD, \qquad (5)$$
$$\forall t \geq 2, 1 \leq k \leq t-1, v \in \mathcal{L}(t),$$

where $L_{(v_1,t_1)}^{(v_2,t_2)}$ *is the* **maximum delay** *among all paths from* (v_1,t_1) *to* (v_2,t_2), *defined as:*

$$L_{(v_1,t_1)}^{(v_2,t_2)} = \max_{(\mathbf{v}_{t_1}, \mathbf{v}_{t_1+1}, ..., \mathbf{v}_{t_2}) \in \Gamma_{(v_1,t_1)}^{(v_2,t_2)}} \sum_{i=1}^{t_2-t_1} C(\mathbf{v}_{t_1+i}). \qquad (6)$$

PROOF. See Appendix B.1.

A simple upper-bound on the maximum difference in starting times follows immediately from Lemma 5:

Lemma 6. *Suppose* $\max_{u,v \in \mathcal{L}(1)} |S(u,1) - S(v,1)| \leq \gamma$ *a.s. for some* $0 < \gamma < \infty$. *Then for all* $t > 1$

$$\max_{u,v \in \mathcal{L}(t)} |S(u,t) - S(v,t)| \leq$$

$$\gamma + \max_{u,v \in \mathcal{L}(t)} \max_{u',v' \in \mathcal{L}(1)} \left|L_{(u',1)}^{(u,t)} - L_{(v',1)}^{(v,t)}\right| \text{ a.s.} \qquad (7)$$

---

[4]Technically, $d_{(v_1,t_1)}(t_2)$ is the "set of indices corresponding to the subset of $V_P(t_2)$ to which a path from (v_1,t_1) exists". Since it does not cause any confusion, we use the more concise statement. The same applies to the ancestor set.

In particular, when $C(v,t) \leq \delta$ a.s. $\forall t \in \mathbb{N}, v \in \mathcal{L}(t)$, the above inequality implies

$$\max_{u,v \in \mathcal{L}(t)} |S(u,t) - S(v,t)| \leq \gamma + 2(t-1)\delta \text{ a.s.} \quad (8)$$

PROOF. See Appendix B.2.

Relating to the definition of a synchronized schedule (Definition 3), we can use Equation (8) as an easy upper-bound for the transient state, by letting $t = T(\mathcal{G}_P)$. This bound, however, does not work for the steady state, since it grows as t increases. To obtain a uniform bound on the maximum differences in starting times for the steady state, we will need more structure on the precedence graph.

We say that two levels of the precedence graph, $V_P(t_1)$ and $V_P(t_2)$, $t_1 < t_2$, are **totally connected (TC)**, if there exists at least one path between *any vertex* in $V_P(t_1)$ and *any vertex* in $V_P(t_2)$, i.e.

$$\Gamma_{(v_1,t_1)}^{(v_2,t_2)} \neq \emptyset, \forall v_1 \in \mathcal{L}(t_1), v_2 \in \mathcal{L}(t_2).$$

The following theorem shows we can achieve a much better upper-bound for the differences in starting times among vertices at level $V_P(t)$, if $V_P(t)$ is totally connected with $V_P(t + N(t))$ for some $N(t) < \infty$. This upper bound is similar to that in Lemma 4.4 in [2], while in [2] the bound is for the difference of starting times between two vertices at *adjacent* levels.

Theorem 7 (Upper-bound for Maximum Difference in Starting Times). *Suppose for some $t \in \mathbb{N}$*

$$N(t) := \inf \{n \in \mathbb{N} : V_P(t) \text{ is TC with } V_P(t+n)\} < \infty. \quad (9)$$

Then,

$$\max_{u,v \in \mathcal{L}(t+N(t))} |S(u, t + N(t)) - S(v, t + N(t))|$$
$$\leq \max_{u,v \in \mathcal{L}(t+N(t))} \max_{l \in \mathcal{L}(t)} \left| L_{(l,t)}^{(u,t+N(t))} - L_{(l,t)}^{(v,t+N(t))} \right|, \quad (10)$$

In particular, when $C(v,t) \leq \delta$ a.s. $\forall t \in \mathbb{N}, v \in \mathcal{L}(t)$, the above inequality implies

$$\max_{u,v \in \mathcal{L}(t+N(t))} |S(u, t + N(t)) - S(v, t + N(t))| \leq 2N(t)\delta. \quad (11)$$

PROOF. See Appendix B.3.

A consequence of Theorem 7 is as follows: if two levels $V_P(t_1)$ and $V_P(t_2)$, $t_1 < t_2$, are totally connected, the difference in starting times of $V_P(t_2)$ can be bounded *independently* from the starting times at $V_P(t_1)$, by a function of *only* the sequence of random delays $\{C(v,t)\}$ between t_1 and t_2. This result may seem quite counter-intuitive, as one may expect perturbations in starting times in $V_P(t_1)$ to impact on all subsequent levels. It turns out that the *total connectedness* is the key: if some vertex in $V_P(t_1)$ has a significantly more delayed starting time compared to other vertices in $V_P(t_1)$, the total connectedness between the two levels allows for such a discrepancy in starting time to "propagate" and eventually delay *all* vertices in $V_P(t_2)$, and hence "synchronize" the starting times in $V_P(t_2)$. Our design of self-synchronizing scheduling via locally enforced schedules was essentially inspired by this intuition of "synchronization by delay propagation".

Remark (Triviality of Transient State Upper-bound). *Equation (8) in Lemma 6 essentially suggests that the upper-bound for the transient state (Equation (2) in Definition 3) is trivial once a finite $T(\mathcal{G}_P)$ is found. Therefore, from this point on, we will omit the statement/proof for the transient state, if we have already shown the existence of $T(\mathcal{G}_P)$ and the uniform upper-bound for the steady state.*

5. NECESSARY AND SUFFICIENT CONDITIONS FOR SYNCHRONIZATION

We prove in this section a necessary and sufficient condition for a MIS schedule $\mathcal{L}$ to be synchronized, which is closely related to the notion of total connectedness, introduced in Section 4. A practical implication of this is immediate: If a wireless network is currently run using some slotted TDMA schedule, it is possible that the *same* schedule can be implemented *without* an explicit slotted mechanism, which relies on expensive clock synchronization, by simply using a locally enforced protocol similar to that described in Subsection 3.3. Theorem 9 and Proposition 12 below ensure this possibility can be verified for any schedule and contention graph, in a finite number of steps.

We begin with a simple lemma, which states that on the precedence graph once the descendant set of a vertex in $V_P(t_1)$ includes all of $\mathcal{L}(t_2)$ for some $t_2 > t_2$, it will continue to include all of $\mathcal{L}(t'), \forall t' \geq t_2$. The proof follows from the nice property that each slot of $\mathcal{L}$ is a maximal independent set of $\mathcal{G}_I$.

Lemma 8. *Let $t_2 > t_1$. If $d_{(v,t_1)}(t_2) = \mathcal{L}(t_2)$ for some $v \in \mathcal{L}(t_1)$, then $d_{(v,t_1)}(t') = \mathcal{L}(t')$ for all $t' \geq t_2$.*

PROOF. Since $\mathcal{L}(t)$ is a maximal independent set of $\mathcal{G}_I$ for all $t \in \mathbb{N}$, for every $v_{t+1} \in \mathcal{L}(t+1)$, there exists at least one vertex $v_t \in \mathcal{L}(t)$ such that there is an edge from (v_t, t) to $(v_{t+1}, t+1)$ in the precedence graph $\mathcal{G}_P$. Therefore, once the descendant set $d_{(v,t_1)}(t_2)$ is equal to $\mathcal{L}(t_2)$ at slot t_2, it will continue to include all of $\mathcal{L}(t_2 + 1)$. Repeating this argument, we have that $d_{(v,t_1)}(t')$ includes all of $\mathcal{L}(t')$ for all $t' > t_2$. $\square$

Theorem 9 provides a necessary and sufficient condition for an MIS schedule to be synchronized.

Theorem 9. *An MIS schedule $\mathcal{L}$ is synchronized if and only if the following holds: Pick **any** [5] $t \in \mathbb{N}$*

$$\sup_{v \in \mathcal{L}(t)} \inf \{\delta \in \mathbb{N} : d_{(v,t)}(t+\delta) = \mathcal{L}(t+\delta)\} < \infty \quad (12)$$

where $d_{(v,t)}(t+\delta)$ is the set of descendants of vertex (v,t) in the future slot $t + \delta$, as defined in Definition 4.

We defer the proof of the theorem till the end of the section. Theorem 9 basically states that a schedule is synchronized if and only if for *any* vertex $v \in \mathcal{L}(t)$, its descendant set $d_{(v,t)}(t+\delta)$ will include all of $\mathcal{L}(t+\delta)$ for some finite δ. In light of Lemma 8, Equation (12) is essentially the same as saying that $\mathcal{L}(t)$ totally connects with $\mathcal{L}(t+N)$ for some finite N. In fact,

$$\sup_{v \in \mathcal{L}(t)} \inf \{\delta \in \mathbb{N} : d_{(v,t)}(t+\delta) = \mathcal{L}(t+\delta)\} = N(t), \quad (13)$$

---

[5] Turns out it suffices to pick only one slot to check the necessary and sufficient condition. Details are given in Appendix A.

where $N(t)$ is defined as in Equation (9).

We will however stick with Equation (12) as it is more explicit than the totally connectedness statement, and as will be shown below, it leads to simple algorithms to check for synchronization for a given MIS schedule.

The sufficiency of (12) is not a complete surprise, as the total connectedness requirement in Theorem 7 can be readily extended to that of Equation (13) to become a sufficient condition for synchronization (More details are given in the proof for Theorem 9). The additional result provided by Theorem 9 is that this condition is also necessary.

While intuitive, the necessary condition is not trivial to verify: If we were to directly use the condition in Equation (12) to check for total connectedness starting from $\mathcal{L}(t)$ for some $t \in \mathbb{N}$, by following the evolution of descendant sets for all $v \in \mathcal{L}(t)$, it is unclear how far down the precedence graph one needs to examine before one could *stop* and declare that the schedule is not synchronized.

It turns out there is a simple algorithm that is guaranteed to complete in a finite number of steps by checking the existence of an **absorbing subset** of the precedence graph. This concept of absorbing subset will also be used in proving the necessary condition in Theorem 9.

Definition 10. *Consider the precedence graph $\mathcal{G}_P$ induced by schedule $\mathcal{L}$ and contention graph $\mathcal{G}_I$. $C_a \subset \mathcal{L}(t)$ is said to be an **absorbing subset** of $\mathcal{L}(t)$, with period j_a if the following conditions hold:*
1. $\bigcup_{v \in C_a} d_{(v,t)}(t + j_a K) = C_a$,
2. $\bigcup_{v \in C_a} d_{(v,t)}(t + i) \neq \mathcal{L}(t + i), \forall 1 \leq i < j_a K$,
where $K = |L|$ is the length of the scheduling frame.

In other words, the union of descendant sets starting from vertices in C_a at time t becomes again C_a in slot $t + j_a K$, *and* this union never covers all of $\mathcal{L}(t')$ for all $t < t' < t + j_a K$.

Observe that since the directed edges in the $\mathcal{G}_P$ only exist between adjacent levels, the sequence of descendant sets from vertex (v,t) at time t, $\left\{d_{(v,t)}(t+i)\right\}_{i \in \mathbb{N}}$, has a Markovian property, in the sense that $\forall m \geq 1, \left\{d_{(v,t)}(t+i)\right\}_{i \geq m}$ is fully determined by $d_{(v,t)}(t+m)$. The following lemma follows from this observation.

Lemma 11. *If C_a is an absorbing subset of $\mathcal{L}(t)$ for some $t \geq 1$, then $\mathcal{L}$ is not synchronized.*

PROOF. See Appendix B.4.

Lemma 11 hints towards a simple stopping criterion for checking synchronization: whenever the emergence of an absorbing subset is observed, the checking algorithm can be terminated and $\mathcal{L}$ will be declared as not synchronized. This translates into the following algorithm:

Synchronization Checking Algorithm

Pick $t \in \mathbb{N}$
for every $v \in \mathcal{L}(t)$ **do**
  j=1
  while $d_{(v,t)}(t+jK) \neq \mathcal{L}(t+jK)$ (*) **do**
    if $\exists i, 1 \leq i < j$, so that $d_{(v,t)}(t+iK) = d_{(v,t)}(t+jK)$ (**) **then**
      return "$\mathcal{L}$ is not synchronized"
    end if
    j++;
  end while
end for
return "$\mathcal{L}$ is synchronized"

Condition (*) checks whether the current descendant set already includes all nodes in the scheduling slot, and condition (**) checks for the emergence of an absorbing subset. The following proposition states that the above algorithm finishes in finite time, given that the contention graph $\mathcal{G}_I$ is finite.

Proposition 12. *If we follow the sequence of descendant sets $\left\{d_{(v,t)}(t+jK)\right\}_{j=1}^{\infty}$ of a vertex $v \in \mathcal{L}(t)$, one of the following two is guaranteed to happen after finite $j, 1 \leq j \leq 2^{|\mathcal{L}(t)|}$:*
1. $d_{(v,t)}(t + jK) = \mathcal{L}(t + jK)$
2. $d_{(v,t)}(t + jK)$ is absorbing subset of $\mathcal{L}(t + jK)$.
This implies that the synchronization checking algorithm terminates in a finite number of steps for any MIS schedule $\mathcal{L}$ and contention graph $\mathcal{G}_I$.

PROOF. See Appendix B.5.

While the worst-case runtime for the above algorithm is exponential with respect to the number of vertices, we point out that in reality the algorithm is *not* repeatedly performed by each wireless node, but is instead run *only once* to check an MIS schedule for self-synchronization.

We are now ready to prove Theorem 9.

PROOF (THEOREM 9). (**Sufficient**) Suppose there exists a finite number M such that for all $t \in \mathbb{N}$ and all $v \in \mathcal{L}(t)$,

$$\inf\left\{\delta : d_{(v,t)}(t+\delta) = \mathcal{L}(t+\delta)\right\} < M.$$

From Lemma 8, we have:

$$d_{(v,t)}(t + j^*K) = \mathcal{L}(t + j^*K), \forall v \in \mathcal{L}(t),$$

where $j^* = \inf\{j : jK \geq M\}$, so $\mathcal{L}(t)$ is totally connected with $\mathcal{L}(t + j^*K)$. As the scheduling frame L is repeated in $\mathcal{L}$, Lemma 18 in Appendix A implies that this total connectedness also extends to all other slots so that: $\mathcal{L}(t')$ is totally connected with $\mathcal{L}(t' + (j^* + 1)K)$, $\forall t' \in \mathbb{N}$. The steady state upper-bound (Equation (3)) in the definition of synchronization is therefore proved by applying Equation (11) in Theorem 7. The transient state upper-bound (Equation (2)), as commented in the remark at the end of Section 4, follows from Equation (8), by noting that $T(\mathcal{G}_P) = 1 + (j^* + 1)K$.

(**Necessary**) Suppose there exists $v \in \mathcal{L}(t)$ such that

$$\inf\left\{\delta : d_{(v,t)}(t+\delta) = \mathcal{L}(t+\delta)\right\} = \infty. \qquad (14)$$

We would like to show there exist some bounded delay distributions that will lead to unbounded difference in starting times as $t \to \infty$. From Proposition 12, Equation (14) implies that there exists at least one absorbing subset in $\left\{d_{(v,t)}(t+jK)\right\}_{j \geq 1}$. Suppose $d_{(v,t)}(t')$ is an absorbing subset of $\mathcal{L}(t')$ for some $t' > t$, with period j_a. By Definition 10 of an absorbing subset, $d_{(v,t)}(t' + m)$ is a strict (or proper) subset of $\mathcal{L}(t' + m)$ for all $1 \leq m \leq j_a K$. For some $i \in \mathbb{N}$, denote by $\Delta(i)$ the set of vertices in $\mathcal{V}_P$ that do not belong to the *union of descendant sets* of (v,t) from $d_{(v,t)}(t')$ to $d_{(v,t)}(t' + i(j_a K))$:

$$\Delta(i) := \bigcup_{m=1}^{i(j_a K)} \{(w, t' + m) : w \in \mathcal{L}(t' + m)\}$$
$$- \bigcup_{m=1}^{i(j_a K)} \{(u, t' + m) : u \in d_{(v,t)}(t' + m)\}.$$

Fix $\delta > 0$, for all $i \in \mathbb{N}$, assign all vertices in $\Delta(i)$ a deterministic delay of $C(v,t) = \delta$, and assign all vertices in $\bigcup_{m=1}^{i(j_a K)}\{(u, t'+m) : u \in d_{(v,t)}(t'+m)\}$ a deterministic delay of $C(v,t) = \delta$.

For any i, pick two vertices $p \in d_{(v,t)}(t'+iK)$ and $q \in \mathcal{L}(t'+iK)\backslash d_{(v,t)}(t'+iK)$. Notice that by construction, all of p's ancestors are in $\Delta(i)$ but none of q's ancestors. The following inequality follows from the constructed discrepancy in random delay distributions. Using the decomposition of $S(v,t)$ from Lemma 5 (Equation (5)), we have:

$$
\begin{aligned}
\big| S(p, t' + iK) &- S(q, t' + iK) \big| = \\
&\Bigg| \max_{m \in a_{(p,t'+iK)}(t')} \left\{ S(m,t') + L_{(m,t')}^{(p,t'+iK)} \right\} \\
&\quad - \max_{n \in a_{(q,t'+iK)}(t')} \left\{ S(n,t') + L_{(n,t')}^{(q,t'+iK)} \right\} \Bigg| \\
= &\Bigg| \max_{m \in a_{(p,t'+iK)}(t')} \left\{ S(m,t') \right\} + 2iK\delta \\
&\quad - \max_{n \in a_{(q,t'+iK)}(t')} \left\{ S(n,t') \right\} - iK\delta \Bigg| \\
\geq &\; iK\delta - \max_{m,n \in \mathcal{L}(t')} \big| S(m,t') - S(n,t') \big|.
\end{aligned}
$$

Hence for all $\delta > 0$, $\lim_{i \to \infty} |S(p, t'+iK) - S(q, t'+iK)| = \infty$ a.s.. This means upper-bound for the steady state (Equation (3)) in Definition 3 does not exist, and $\mathcal{L}$ is therefore not synchronized. $\qquad\square$

6. EXISTENCE OF SYNCHRONIZED MIS SCHEDULES ON GENERAL GRAPHS

It is important to note that the necessary condition presented in Theorem 9 is *not trivial*, in the sense not *all* MIS schedules that satisfy Definition 1 are synchronized. Subsection 7.1 shows a simple counter-example. If this is the case, it is then natural to question whether there exists *any* synchronized MIS schedule for a given contention graph $\mathcal{G}_I$. Fortunately, the main result in this section shows that there exists at least one synchronized MIS schedule for *any* connected contention graph.

We begin with a useful corollary of Theorem 7, which shows how the degree of synchronization scales with the number of slots it takes for two levels to be totally connected in an induced precedence graph.

Corollary 13. *Consider a synchronized schedule $\mathcal{L}$ on $\mathcal{G}_I$ and the corresponding induced precedence graph $\mathcal{G}_P$. If for some $t' \in \mathbb{N}$, there exists $j \in \mathbb{N}$ such that*

$$\mathcal{L}(t') \text{ is totally connected with } \mathcal{L}(t'+jK), \quad (15)$$

where $K = |L|$ is the length of the scheduling frame. Then for all $t \geq 1 + (j+1)K$ we have:

$$\max_{v_1, v_2 \in \mathcal{L}(t)} |S(v_1, t) - S(v_2, t)| \leq 2\delta(j+1)K \text{ a.s.,}$$

whenever the random delays $|C(v,t)| \leq \delta$ a.s. for all $t \geq 1, v \in \mathcal{L}(t)$.

PROOF. See Appendix B.6.

We define the notion of a covering closed walk on the contention graph:

Definition 14. *Given an (undirected) connected contention graph $\mathcal{G}_I = (\mathcal{V}_I, \mathcal{E}_I)$, a **closed walk** on $\mathcal{G}_I$ is a sequence of nodes, $Q = (q_1, q_2, ..., q_R)$, such that*

$$
\begin{aligned}
&(q_i, q_{i+1}) \in \mathcal{E}_I, \forall 1 \leq i \leq R-1 \\
&q_1 = q_R
\end{aligned}
$$

*Q is said to be a **covering closed walk** of $\mathcal{G}_I$ if in addition Q covers all nodes in $\mathcal{G}_I$, i.e. $\bigcup_{q_i \in Q} q_i = \mathcal{V}_I$.*

We now state the main theorem.

Theorem 15 (Existence of Synchronized Schedule in General Graphs). *Given any connected contention graph $\mathcal{G}_I$, there exists a finite MIS scheduling frame $L = (L_1, L_2, ..., L_K)$ such that the corresponding schedule $\mathcal{L}$ is synchronized.*

Moreover, one can find an $\mathcal{L}$ so that for all $t \geq 4|Q| - 1$:

$$\max_{v_1, v_2 \in \mathcal{L}(t)} |S(v_1, t) - S(v_2, t)| \leq 4\delta(2|Q|-1) \text{ a.s.,}$$

*where Q is a **covering closed walk** on $\mathcal{G}_I$ that covers all nodes of $\mathcal{G}_I$ [6], whenever $|C(v,t)| \leq \delta$ a.s. $\forall t \in \mathbb{N}, v \in \mathcal{L}(t)$,.*

We offer a constructive proof to Theorem 15, which is divided into two parts: We first give the construction of a schedule $\mathcal{L}$, and proceed to prove that $\mathcal{L}$ is indeed synchronized and achieves the upper-bound. The construction phase can also be readily extended to serve as a simple algorithm for finding a synchronized schedule given an contention graph.

PROOF (THEOREM 15). (**Construction of $\mathcal{L}$**) Let $Q = (q_1, q_1, ..., q_R)$ be a covering closed walk on $\mathcal{G}_I$. Let L be a scheduling frame,

$$L = (L_1, L_2, ..., L_{2R-1}),$$

where $R = |Q|$. Choose the first slot $L_1 = \left\{ v_1^1, v_2^1, ..., v_M^1 \right\}$ to be *any* maximal independent set of $\mathcal{G}_I$ that contains the node q_1, which can be easily generated by a greedy algorithm, and label each v_i^1 in accordance with their order of appearance in the walk Q, so that v_i^1 is the i'th element of L_1 that appears in Q. If $v_i^1 \in L_1$ appears more than once in Q, only its first appearance in Q is regarded when labeling. Let $t(i)$ be the first time v_i^1 appears in the sequence Q, i.e.

$$t(i) = \min_j \{j : q_j = v_i^1\}. \quad (16)$$

The above mentioned labeling of L_1 implies that $t(m) < t(n)$ whenever $1 \leq m < n \leq M$.

Now that we have L_1, construct the first R slots of L as follows: For all $i \in \{2, 3, ..., R\}$,

1. If $q_i \in L_1$, let $L_i = L_1$.

2. Otherwise, let L_i be *any* maximal independent set that contains q_i and all nodes in L_1 that are not in the neighborhood of q_i, i.e.

$$L_i \supset \{q_i\} \cup \{v \in L_1 : v \notin \mathcal{N}_I(q_i)\}. \quad (17)$$

Note that the right-hand side of Equation (17) is indeed an independent set, so L_i can again be found by simple greedy algorithms[7].

---

[6] Note that the elements of Q are *not* assumed to be unique. Hence such a covering closed walk exists for any connected graph. We assume finding such a Q is not too difficult, and can be performed by some separate algorithm.

[7] In fact, one way to find such an L_i is to check all nodes that are neighbors with nodes in $\{v \in L_1 : v \in \mathcal{N}_I(q_i)\}$, and add those who are *not* neighbors with nodes in $q_i \cup \{v \in L_1 : v \notin \mathcal{N}_I(q_i)\}$.

In particular, note that since Q is a closed walk ($q_R = q_1$), we have $L_R = L_1$ due to Case 1 in the above construction. We complete the scheduling frame L by letting the second part of L be the reverse of the first part, so that

$$(L_{R+1}, L_{R+2}, ..., L_{2R-1}) = (L_{R-1}, L_{R-2}, ..., L_1).$$

By the above construction, L_i is a maximal independent sets of $\mathcal{G}_I$ for all $1 \le i \le 2R-1$. Observe also since Q covers $\mathcal{V}_I$ and $q_i \in L_i$, the union of the first R slots $\bigcup_{1 \le i \le R} L_i$ also covers $\mathcal{V}_I$. Both the fairness and MIS conditions in Definition 1 are satisfied. Therefore, the schedule $\mathcal{L}$ generated by repeating L is a valid MIS schedule.

(**Synchronization of $\mathcal{L}$**) We argue that $\mathcal{L}$, as constructed above, is synchronized. Consider again the precedence graph $\mathcal{G}_P = (\mathcal{V}_P, \mathcal{E}_P)$ induced by $\mathcal{L}$ and $\mathcal{G}_I$. It suffices to show that $\mathcal{L}(1)$ totally connects with $\mathcal{L}(1+K)$, where $K = |L| = 2R - 1$. We can then invoke Corollary 13 to show synchronization and obtain the corresponding upper-bound.

The following two observations are useful:

- (**Observation 1**) If $v_i^1 \in \mathcal{L}(t)$ for some $2 \le t \le R$, then it must be that

$$\Gamma_{(v_i^1, 1)}^{(v_i^1, t)} \ne \emptyset.$$

  That is, there exists a path from $(v_i^1, 1)$ to (v_i^1, t) on the precedence graph for all $1 \le i \le M$ ($M = |L_1|$), or equivalently,

$$v_i^1 \in a_{(v_i^1, t)}(1).$$

  This is true because either v_i^1 or at least one of v_i^1's neighbors is scheduled in every slot $\mathcal{L}(t')$ for all $1 \le t' \le t$ based on the construction of $(L_2, L_3, ..., L_R)$. These paths are illustrated by the vertical dotted arrows in Figure 3.

- (**Observation 2**) We have

$$q_i \in d_{(q_{i-1}, i-1)}(i), \forall 2 \le i \le R.$$

  This is because node q_i is guaranteed to be in slot $\mathcal{L}(i)$ by construction, and q_i and q_{i-1} are neighbors by the definition of a walk ($(q_i, q_{i+1}) \in \mathcal{E}_I$). This further implies that the sequence of vertices in the precedence graph $\mathcal{G}_P$,

$$((q_1, 1), (q_2, 2), ..., (q_i, i)), 1 \le i \le R$$

  is a *path* in $\mathcal{G}_P$. This path is illustrated by the concrete diagonal arrows that traverse across the precedence graph in Figure 3.

The rest of the proof will rely on following intuition: the path of $\{(q_t, t)\}$ will *intersect* with the paths of $\{(v_i^1, t)\}$ for all $1 \le i \le M$. Hence, the delay information in the precedence graph is essentially "propagated" by the path of $\{(q_t, t)\}$ and "saved" by the paths of $\{(v_i^1, t)\}$. We now formalize this intuition.

Recall the quantity $t(i)$ defined in Equation (16) in the first part of the proof. Relating to the above intuition, the sequence of $\{t(i)\}$ essentially marks the slots where the path of $\{(q_t, t)\}$ and $\{(v_i^1, t)\}$ is guaranteed to intersect (See Figure 3). Since $v_i^1 = q_{t(i)} \in \mathcal{L}(t), \forall 1 \le i \le M$, combining both observations above, we claim that:

$$\{v_1^1, v_2^1, ..., v_i^1\} \subset a_{(v_i^1, t(i))}(1), \forall 1 \le i \le M \qquad (18)$$

The claim can be proved by a simple induction. It is true when $i = 2$. Indeed, from Observation 1, we have $v_2^1 \in a_{(v_2^1, t(2))}(1)$, and from Observation 2, we have $v_1^1 = q_{t(1)} \in a_{(v_2^1, t(2))}(1)$. Suppose Equation (18) is true for $i = j-1 \ge 2$. We would like to show that it also holds for $i = j$. From Observation 1, we again have $v_j^1 \in a_{(v_j^1, t(j))}(1)$. Also, $v_{j-1}^1 = q_{t(j-1)} \in a_{(v_j^1, t(j))}(t(j-1))$ by Observation 2. Combining with induction hypothesis on $j - 1$, we get:

$$\{v_1^1, v_2^1, ..., v_j^1\} = a_{(v_{j-1}^1, t(j-1))}(1) \cup \{v_j^1\} \subset a_{(v_j^1, t(j))}(1),$$

which proves the claim.

Let us now look at slot $\mathcal{L}(R)$. Note that $\mathcal{L}(R) = L_R = L_1 = (v_1^1, v_2^1, ..., v_M^1)$ by construction. Equation (18) combined with Observation 1 implies that the set of ancestors of (v_i^1, R) in slot $\mathcal{L}(1)$ includes all $v_j, j \le i$ (Figure 3), i.e.: For all $1 \le i \le M$,

$$a_{(v_i^1, R)}(1) \supset \{v_1^1, v_2^1, ..., v_i^1\}.$$

In particular, let $i = M$, we have that there exists at least one path from vertex $(v_j^1, 1)$ to (v_M^1, R), for all $1 \le j \le M$, i.e.

$$\Gamma_{(v_j^1, 1)}^{(v_M^1, R)} \ne \emptyset, \forall 1 \le j \le M. \qquad (19)$$

Using essentially the same arguments, we get a symmetric result in $\mathcal{L}(2R - 1)$: For all $1 \le i \le M$,

$$d_{(v_i^1, R)}(2R - 1) \supset \{v_1^1, v_2^1, ..., v_i^1\}.$$

In particular, let $i = M$, we have

$$\Gamma_{(v_M^1, R)}^{(v_j^1, 2R-1)} \ne \emptyset, \forall 1 \le j \le M. \qquad (20)$$

Combining (19) and (20), we have that $\mathcal{L}(1)$ **totally connects** with $\mathcal{L}(2R - 1)$, i.e.:

$$d_{(v_i^1, 1)}(2R - 1) = \mathcal{L}(2R - 1), \forall 1 \le i \le M.$$

By Lemma 8, $\mathcal{L}(1)$ also totally connects with $\mathcal{L}(2R)$. Note that $2R = 1 + 2|Q| - 1 = 1 + K$. Therefore, $\mathcal{L}$ is synchronized, and the upper-bound in the theorem (Equation (16)) follows from Corollary 13. $\qquad \square$

7. EXAMPLES AND APPLICATIONS

7.1 Good and Bad MIS Schedules

We show here, by a simple counter-example, that the necessary condition proven in Theorem 9 is not trivial, in the sense that not all MIS schedules satisfying conditions in Definition 1 are synchronized. Consider the *6-node-ring* contention topology depicted in Figure 1. Consider the following two scheduling fames:
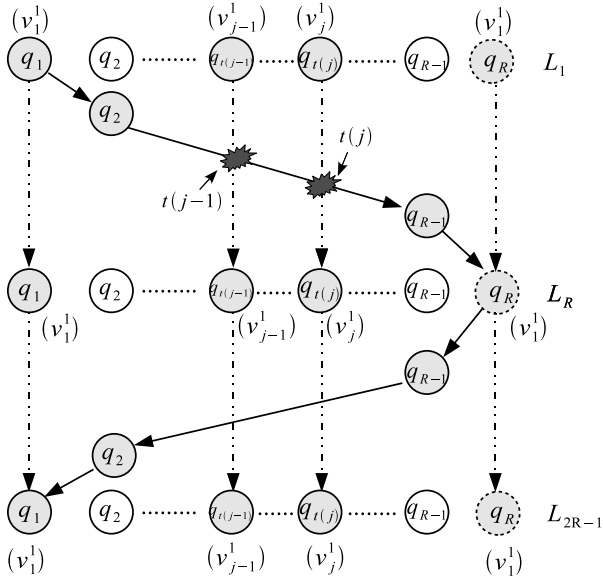
1. Scheduling Frame A, $|L_A| = 2$,

$$L_{A1} = \{1, 3, 5\}, L_{A2} = \{2, 4, 6\}.$$

2. Scheduling Frame B, $|L_B| = 3$,

$$L_{B1} = \{1, 4\}, L_{B2} = \{2, 5\}, L_{B3} = \{3, 6\}.$$

Denote by $\mathcal{L}_A$ (and respectively, $\mathcal{L}_B$) the MIS schedule generated by repeating L_A (L_B). One can check that both schedules contain only maximal independent sets of the contention graph. For $\mathcal{L}_B$, every node that is not transmitting

Figure 3: A (high-level) illustration of the precedence graph induced by $\mathcal{L}$ in Theorem 15. The gray nodes represent nodes that are scheduled in the corresponding slot of $\mathcal{L}$. Node v_1^1 (same as q_1) are replicated (right-most column) for the easiness of presentation.

in a slot has exactly one neighbor who is transmitting. Also, both schedules cover all six nodes of the contention graph. Therefore, both $\mathcal{L}_A$ and $\mathcal{L}_B$ are legitimate MIS schedules by Definition 1. However, we have the following proposition:

Proposition 16. $\mathcal{L}_A$ *is synchronized but* $\mathcal{L}_B$ *is NOT synchronized.*

PROOF. The proof is simple once we construct the precedence graphs for both schedules (Figures 4 and 5). The synchronization property of $\mathcal{L}_A$ follows from (Figure 4):

$$V_P^A(t) \text{ totally connects with } V_P^A(t+2), \forall t \geq 1.$$

Hence $\mathcal{L}_A$ is synchronized by Theorem 9. The fact that $\mathcal{L}_B$ is not synchronized can be easily seen from Figure 5. Starting from vertex 1 in $V_P^B(1)$, its sequence of descendant sets never reaches all vertices in a given slot of concurrent transmissions, until it comes back to itself, $\{1\}$, in $V_P^B(7)$ (Figure 5). Therefore, $\{1\}$ forms an absorbing subset of V_P^B with a period of $j_a = 6/3 = 2$ by Definition 10. $\mathcal{L}_B$ is hence not synchronized by Lemma 11. $\square$

The reason why $\mathcal{L}_B$ is not synchronized is intuitive: every node in $\mathcal{G}_I$ only hears its left neighbor to decide when to transmit next. Therefore, due to the ring topology, two separate paths of "delay propagation" form and they never intersect.

Simulations: Figure 6 compares two sample paths of the maximum difference in starting times, $\max_{u,v \in \mathcal{L}(t)} |S(u,t) - S(v,t)|$, when Schedule A and Schedule B are used. The random delays $C(v,t)$ are i.i.d and is distributed uniformly in $[0,1]$. As expected, the maximum difference in starting times for Schedule A (synchronized) is bounded closely to zero, that of Schedule B (not synchronized) varies drastically. Figure 7 compares the variance of the maximum difference in starting times by using two schedules, computed
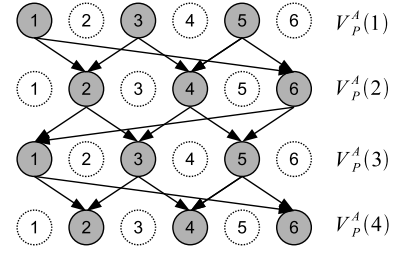
Figure 4: Precedence graph representation for $\mathcal{L}_A$ (synchronized).
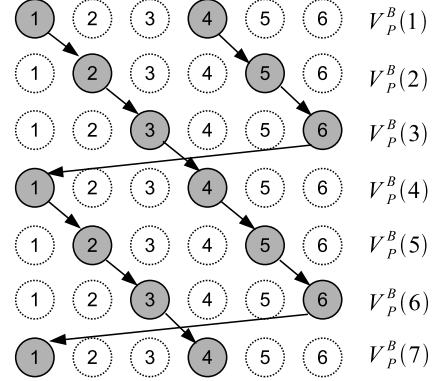
Figure 5: Precedence graph representation for $\mathcal{L}_B$ (not synchronized).

over 600 trials. Again, Schedule A retains a small variance, while Schedule B leads to a large and steadily increasing variance.

7.2 Linear Multi-hop Wireless Networks with Two-Hop Interference

In this subsection, we apply the theory developed in the previous sections to study a self-synchronized MIS schedule for a linear multi-hop wireless network where each node interferes with its two-hop neighbors. While our results apply to any general contention topology, we chose this example because it captures the phenomenon of multi-hop interference, which is present in almost all real-world wireless networks, while being simple enough to obtain closed-form bounds on the maximal difference in starting times (Proposition 17).

Consider a network of M wireless nodes aligned in a line topology, spaced equally at a distance of l apart (Figure 8). Every node interferes with all nodes that are within a distance of $2l$ (two hops). Labeling the nodes as $1, 2, ..., M$ from left to right with respect to the physical topology, we can construct the *contention graph* $\mathcal{G}_I$ as depicted in the lower plot in Figure 8. The two-hop interference constraint is equivalent to:

$$\mathcal{N}_I(t) = \{t-2, t-1, t, t+1, t+2\}, \forall 3 \leq t \leq M-2$$
$$\mathcal{N}_I(1) = \{1, 2, 3\}, \mathcal{N}_I(2) = \{1, 2, 3, 4\},$$
$$\mathcal{N}_I(M-1) = \{M-3, M-2, M-1, M\},$$
$$\mathcal{N}_I(M) = \{M-2, M-1, M\}.$$
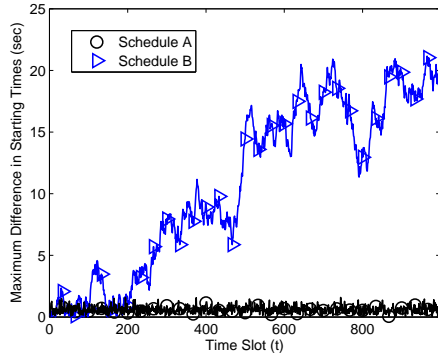
For the moment assume the total number of nodes is a

Figure 6: Sample path comparison.

Figure 7: Variance comparison.

Figure 8: Contention graph for a linear wireless network with two-hop interference.

multiple of 3 ($M = 3K$ for some $K \in \mathbb{N}$)[8]. Consider the MIS scheduling frame L, $|L| = 3$, where

$$L_1 = \{1, 4, ..., M - 5, M - 2\}, L_2 = \{2, 5, ..., M - 4, M - 1\},$$
$$L_3 = \{3, 6, ..., M - 3, M\}.$$

Again, let $\mathcal{L}$ be the MIS schedule generated by repeating L. The precedence graph constructed from $\mathcal{L}$ for the case of $M = 9$ is given in Figure 9 (Due to space constraints, we only plot vertices that are included in the slots of concurrent transmissions). We argue that $\mathcal{L}$ is a synchronized schedule and the degree of synchronization is given by the following bound:

Proposition 17. *Consider a linear network of M nodes with two-hop interference. For all $t \geq M - 2$,*

$$\max_{u,v \in \mathcal{L}(t)} |S(u,t) - S(v,t)| \leq 2\delta (M - 3) \ \ a.s.$$

whenever $C(v,t) \leq \delta$ a.s. for all $t \geq 1, v \in \mathcal{L}(t)$.

PROOF. See Appendix B.7.

Proposition 17 shows that the bound on the maximal difference in starting times scales linearly in the total number of nodes (M). This is intuitive, since for any delay from an edge node (node 1 or M) to impact all other nodes in the network, it takes at least M slots due to the linear topology. Similar linear bounds can also be obtained for networks with l-hop interference, where $l \geq 2$ is a positive integer, using essentially the same idea as in the proof of Proposition 17.

8. IMPLEMENTATION ISSUES

The goal of this paper is to establish properties of the self-synchronizing protocol. In this section, we only briefly discuss some preliminary ideas for potential implementations: how a system may be initialized, and how one may handle changes in topologies.

Static Case: Consider a static wireless mesh network, where all nodes stay at a fixed location. The system designer is assumed to know both the interference topology and an already-computed MIS schedule. Such a system undergoes the following two phases:

---

[8]This assumption has very little quantitative impact on the synchronization result we are about to show, but makes the expression much simpler to state.
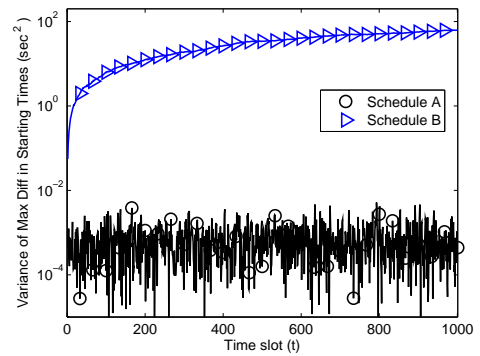
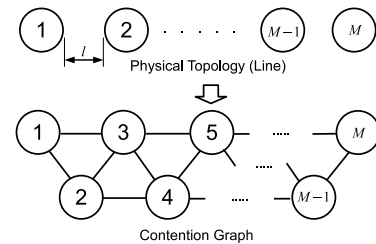*1.(Boot-strapping)* Using a separate communication protocol, nodes communicate with each other to manually synchronize their clocks for the very first frame of their transmissions. If the MIS schedule is not already stored in each node, some central node can also use this phase to distribute the schedule to the rest of the network.
2.(Operating) Once the clocks are synchronized for the first time, the self-synchronization protocol starts, and no more clock synchronization will be needed from this point on.

Dynamic Case: This refers to a fully distributed network, where the contention graph is not initially known to any node, and where the network topology may change over time. Our suggested system setup for the dynamic case incorporates the additional steps of estimation and monitoring, in order to adapt to the changing topology. However, we do not have any theoretical guarantees for the synchronization of a MIS schedule in such a scenario.
1.(Estimating Topology) Using a separate communication protocol, the nodes send probing messages to estimate the interference topology. All information is sent to some central node.
2.(Generating Schedule) The central node computes a synchronizing schedule based on the contention topology and fairness requirements.
3.(Boot-strapping & Operating) Same as Steps *1* and *2* in the static case.
4.(Monitoring) The topology of the network is constantly monitored by individual nodes. If significant changes are detected (through unexpected collisions or other indicators), repeat from Step *1*.
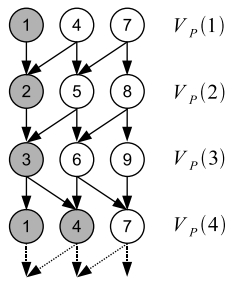
Figure 9: Precedence graph for schedule $\mathcal{L}$ on the linear network.

9. CONCLUSION & OUTLOOK

Finding optimal schedules in wireless multi-hop networks is the object of much current research, and usually relies on some level of explicitly synchronized transmissions. This paper addresses a complementary problem: if the nodes know their scheduling slots, which form maximal independent sets (MIS) of the contention graph, will they keep being self-synchronized? If the answer had been negative, it would mean that some explicit clock synchronization messages would need to be exchanged. Fortunately, we find that even if random but bounded delays add some jitter to the transmission slots, CSMA is able to spontaneously keep transmissions of an MIS schedule in sync, so that the properties of throughput optimality and fairness of a slotted TDMA scheme are preserved without requiring expensive explicit clock synchronization beaconing.

We obtained necessary and sufficient conditions for a MIS schedule to have this nice property (Theorem 9). Some MIS schedules are not self-synchronizing, but we proved that there always exists at least one MIS schedule for every possible connected contention graph (Theorem 15).

Several important questions remain open for future investigation. Firstly, while we have proven the existence of *some* synchronized schedule for a connected graph, it is unclear how to efficiently construct synchronized schedules with quantifiable *performance guarantees* (e.g. throughput, fairness, delays). Secondly, how can one efficiently implement a self-synchronizing schedule in a dynamic network, where the topology may change over time? The current theoretical results are limited to only the static-node setting. Both system experiments or theoretical guarantees for the dynamic setting can be of great interests. Lastly, we assumed in this work that the MIS schedule is periodic and deterministic. In a future work, we would like to address the case where the MIS schedule is not periodic, in order to cover more adaptive scheduling schemes. We conjecture, as early simulations show, that the synchronization effect generalizes to a wide class of more general schedules.

Acknowledgments

10. REFERENCES

[1] M. Alicherry, R. Bhatia, and L. Li. Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks. In *Proc. ACM MobiCom*, page 72. ACM, 2005.

[2] F. Baccelli and Z. Liu. On a class of stochastic recursive sequences arising in queueing theory. *The Annals of Probability*, pages 350–374, 1992.

[3] P. Bjorklund and P. Di Yuan. Resource optimization of spatial TDMA in ad hoc radio networks: A column generation approach. In *Proc. IEEE INFOCOM*, 2003.

[4] G. Brar, D. Blough, and P. Santi. Computationally efficient scheduling with the physical interference model for throughput improvement in wireless mesh networks. In *Proc. ACM MobiCom*, page 13. ACM, 2006.

[5] E. Callaway, P. Gorday, L. Hester, J. Gutierrez, M. Naeve, B. Heile, and V. Bahl. Home networking with IEEE 802.15.4: a developing standard for low-rate wireless personal area networks. *IEEE Communications Magazine*, 40(8):70–77, 2002.

[6] O. Dousse. Revising buffering in multi-hop CSMA/CA wireless networks. In *Proc. Secon*, pages 580–589, San Diego CA, June 2007.

[7] M. Durvy and P. Thiran. A packing approach to compare slotted and non-slotted medium access control. In *Proc. IEEE INFOCOM*, 2006.

[8] S. Ganeriwal, R. Kumar, and M. Srivastava. Timing-sync protocol for sensor networks. In *Proc. ACM Sensys*, pages 138–149. ACM, 2003.

[9] D. Lucarelli and I. Wang. Decentralized synchronization protocols with nearest neighbor communication. In *Proc. ACM Sensys*, pages 62–68. ACM, 2004.

[10] P. Marbach, A. Eryilmaz, and A. Ozdaglar. Achievable rate region of csma schedulers in wireless networks with primary interference constraints. In *Proc. IEEE CDC*, 2007.

[11] R. Mirollo and S. Strogatz. Synchronization of pulse-coupled biological oscillators. *SIAM Journal on Applied Mathematics*, 50(6):1645–1662, 1990.

[12] A. Mutazono, M. Sugano, and M. Murata. Evaluation of robustness in time synchronization for sensor networks. In *Proc. Bionetics'07*, pages 89–92, 2007.

[13] R. Nelson and L. Kleinrock. Spatial TDMA - a collision-free multihop channel access protocol. *IEEE Trans. Comm.*, 33(9):934–944, 1985.

[14] T. Park, T. Kim, J. Choi, S. Choi, and W. Kwon. Throughout and energy consumption analysis of IEEE 802.15.4 slotted CSMA/CA. *IEEE Electronics Letters*, 41(18):1017, 2005.

[15] B. Raman and K. Chebrolu. Revisiting MAC design for an 802.11-based mesh network. In *Proc. 3rd Workshop on Hot Topics in Networks (HotNets-III)*, 2004.

[16] I. Rhee, A. Warrier, M. Aia, J. Min, and M. Sichitiu. Z-MAC: a hybrid mac for wireless sensor networks. *IEEE/ACM Trans. Networking*, 16(3):511–524, 2008.

[17] I. Rhee, A. Warrier, J. Min, and L. Xu. DRAND: distributed randomized TDMA scheduling for wireless ad-hoc networks. In *Proc. ACM MobiHoc*, page 201. ACM, 2006.

[18] L. Roberts. ALOHA packet system with and without slots and capture. *ACM SIGCOMM Computer Communication Review*, 5(2):28–42, 1975.

[19] N. Salem and J. Hubaux. A fair scheduling for wireless mesh networks. In *Proc. WiMesh*, 2005.

[20] S. Singh, P. Acharya, U. Madhow, and E. Belding-Royer. Sticky CSMA/CA: Implicit synchronization and real-time QoS in mesh networks. *Ad Hoc Networks*, 5(6):744–768, 2007.

[21] L. Tassiulas and S. Sarkar. Maxmin fair scheduling in wireless networks. In *Proc. IEEE INFOCOM*, volume 2, pages 763–772, 2002.

APPENDIX

A. UNIFORMITY OF TOTAL CONNECTEDNESS

The reader may have noticed that we pick only one slot $\mathcal{L}(t)$ to check for the sufficient and necessary condition. This is justified by the following lemma, which essentially states: Because the same scheduling frame L is repeated to generate $\mathcal{L}$, the totally connectedness starting from one slot implies that of the others.

Lemma 18. *If for some $t \in \mathbb{N}$, there exists $j \in \mathbb{N}$ such that:*

$$\mathcal{L}(t) \text{ is totally connected with } \mathcal{L}(t + jK),$$

where K is the total number of slots in the scheduling frame L, then

$$\mathcal{L}(t') \text{ is totally connected with } \mathcal{L}(t' + (j+1)K), \forall t' \in \mathbb{N}$$

PROOF. The goal is to show:

$$d_{(v,t')}(t' + (j+1)K) = \mathcal{L}(t' + (j+1)K), \forall v \in \mathcal{L}(t')$$

Assume $t' \neq t$. Let

$$\tilde{i} = \min\{i : t + iK \geq t'\}.$$

Observe that $t + \tilde{i}K < t' + K$. Because each slot of $\mathcal{L}$ is a maximal independent set of $\mathcal{G}_I$, every vertex in $\mathcal{L}(t')$ has at least one descendant in the later slot $\mathcal{L}(t + \tilde{i}K)$ (See also discussion in Section 3.2), i.e.

$$d_{(v,t')}(t + \tilde{i}K) \neq \emptyset, \forall v \in \mathcal{L}(t') \tag{21}$$

Because $\mathcal{L}(t)$ is totally connected with $\mathcal{L}(t + jK)$ and that the scheduling frame L is repeated in $\mathcal{L}$, slot $\mathcal{L}(t + \tilde{i}K)$ is also totally connected with $\mathcal{L}(t + (\tilde{i} + j)K)$, i.e.:

$$d_{(v,t+\tilde{i}K)}(t + (\tilde{i}+j)K) = \mathcal{L}(t + (\tilde{i}+j)K), \forall v \in \mathcal{L}(t + \tilde{i}K)) \tag{22}$$

Combining Equation (21) and Equation (22), we have:

$$d_{(v,t')}(t + (\tilde{i}+j)K) = \mathcal{L}(t + (\tilde{i}+j)K), \forall v \in \mathcal{L}(t + \tilde{i}K)) \tag{23}$$

Since $t + \tilde{i}K < t' + K$, $t + (\tilde{i} + j)K < t' + (j + 1)K$. By Lemma 8, Equation (23) implies

$$d_{(v,t')}(t' + (j+1)K) = \mathcal{L}(t' + (j+1)K), \forall v \in \mathcal{L}(t'),$$

which completes the proof. □

B. PROOFS

B.1 Proof of Lemma 5

PROOF (LEMMA 5). Without any ambiguity, we use $\mathbf{u}, \mathbf{v}$ and $\mathbf{w}$ as shorthand notations for the labels of vertices:

$$\mathbf{v} \sim (v, t) \in V_P(t) \text{ for some } v$$
$$\mathbf{u} \sim (u, t-1) \in V_P(t-1) \text{ for some } u$$
$$\mathbf{w} \sim (w, t-k) \in V_P(t-k) \text{ for some } w$$

The claim is proved by an induction argument on k. In the case of $k = 2$, one can check that the claim is true. Suppose

it is true for some $k - 1 \geq 2$. We have:

$$S(\mathbf{v})$$
$$= \max_{\mathbf{u} \in a_{\mathbf{v}}(t-1)} \{S(\mathbf{u})\} + D + C(\mathbf{v})$$
$$\overset{(a)}{=} \max_{\mathbf{u} \in a_{\mathbf{v}}(t-1)} \left\{ \max_{\mathbf{w} \in a_{\mathbf{u}}(t-k)} \{S(\mathbf{w}) + L_{\mathbf{w}}^{\mathbf{u}}\} + (k-1)D \right\} + D + C(\mathbf{v})$$
$$\overset{(b)}{=} \max_{\mathbf{w} \in a_{\mathbf{v}}(t-k)} \left\{ \max_{\mathbf{u} \in a_{\mathbf{v}}(t-1), \Gamma_{\mathbf{w}}^{\mathbf{u}} \neq \emptyset} \{S(\mathbf{w}) + L_{\mathbf{w}}^{\mathbf{u}}\} \right\} + C(\mathbf{v}) + kD$$
$$= \max_{\mathbf{w} \in a_{\mathbf{v}}(t-k)} \left\{ S(\mathbf{w}) + \max_{\mathbf{u} \in a_{\mathbf{v}}(t-1), \Gamma_{\mathbf{w}}^{\mathbf{u}} \neq \emptyset} \{L_{\mathbf{w}}^{\mathbf{u}} + C(\mathbf{v})\} \right\} + kD$$
$$\overset{(c)}{=} \max_{\mathbf{w} \in a_{\mathbf{v}}(t-k)} \{S(\mathbf{w}) + L_{\mathbf{w}}^{\mathbf{v}}\} + kD,$$

where equality (a) is the induction hypothesis on $k - 1$. Equality (b) is based on the fact that $a_{\mathbf{v}}(t-1)$ and $a_{\mathbf{v}}(t-k)$ are ancestor sets of the same vertex $\mathbf{v}$. Therefore,

$$\{\mathbf{u} \in a_{\mathbf{v}}(t-1) : \Gamma_{\mathbf{w}}^{\mathbf{u}} \neq \emptyset\} \neq \emptyset, \forall \mathbf{w} \in a_{\mathbf{v}}(t-(k+1)).$$

The last equality (c) follows from the definition of $L_{(v_1,t_1)}^{(v_2,t_2)}$ as in Equation (6). Hence the recursive relation also holds for k. □

B.2 Proof of Lemma 6

PROOF (LEMMA 6). By the decomposition of Equation (5),

$$\max_{u,v \in \mathcal{L}(t)} |S(u,t) - S(v,t)| =$$
$$\max_{u,v \in \mathcal{L}(t)} \left| \max_{(u',1) \in a_{(u,t)}(1)} \left\{ S(u',1) + L_{(u',1)}^{(u,t)} \right\} \right.$$
$$\left. - \max_{(v',1) \in a_{(v,t)}(1)} \left\{ S(v',1) + L_{(v',1)}^{(v,t)} \right\} \right|$$
$$\leq \max_{m,n \in \mathcal{L}(1)} |S(m,1) - S(n,1)|$$
$$+ \max_{u,v \in \mathcal{L}(t)} \max_{u',v' \in \mathcal{L}(1)} \left| L_{(u',1)}^{(u,t)} - L_{(v',1)}^{(v,t)} \right|$$
$$\leq \gamma + \max_{u,v \in \mathcal{L}(t)} \max_{u',v' \in \mathcal{L}(1)} \left| L_{(u',1)}^{(u,t)} - L_{(v',1)}^{(v,t)} \right|,$$

which proves Equation (7). Equation (8) follows from the fact that

$$\max_{u,v \in \mathcal{L}(t)} \max_{u',v' \in \mathcal{L}(1)} \left| L_{(u',1)}^{(u,t)} - L_{(v',1)}^{(v,t)} \right| \leq$$
$$2 \max_{u' \in \mathcal{L}(1), u \in \mathcal{L}(t)} L_{(u',1)}^{(u,t)} \leq 2(t-1)\delta,$$

which completes the proof. □

B.3 Proof of Theorem 7

PROOF (THEOREM 7). From Lemma 5, we have:

$$S(u, t + N(t)) - S(v, t + N(t)) =$$
$$\max_{m \in \mathcal{L}(t)} \left\{ S(m,t) + L_{(m,t)}^{(u,t+N(t))} \right\}$$
$$- \max_{n \in \mathcal{L}(t)} \left\{ S(n,t) + L_{(n,t)}^{(v,t+N(t))} \right\}$$
$$\leq \max_{m \in \mathcal{L}(t)} \left\{ S(m,t) + L_{(m,t)}^{(u,t+N(t))} \right.$$
$$\left. - \left(S(m,t) + L_{(m,t)}^{(v,t+N(t))} \right) \right\}$$
$$= \max_{m \in \mathcal{L}(t)} \left(L_{(m,t)}^{(u,t+N(t))} - L_{(m,t)}^{(v,t+N(t))} \right). \tag{24}$$

Similarly,

$$
\begin{aligned}
S(u,t+N(t)) &- S(v,t+N(t)) \geq \\
&- \max_{n\in\mathcal{L}(t)} \left\{ - \left(S(n,t) + L_{(n,t)}^{(u,t+N(t))} \right) \right. \\
&\left. + S(n,t) + L_{(n,t)}^{(v,t+N(t))} \right\} \\
= &\min_{n\in\mathcal{L}(t)} \left(L_{(n,t)}^{(v,t+N(t))} - L_{(n,t)}^{(u,t+N(t))} \right). \quad (25)
\end{aligned}
$$

By combining (24) and (25), we have

$$
\begin{aligned}
|S(u,&t+N(t)) - S(v,t+N(t))| \\
&\leq \max_{l\in\mathcal{L}(t)} \left| L_{(l,t)}^{(v,t+N(t))} - L_{(l,t)}^{(u,t+N(t))} \right|. \quad (26)
\end{aligned}
$$

The claim, (10), follows by maximizing $|S(u,t+N(t)) - S(v,t+N(t))|$ over all $u,v \in \mathcal{L}(t)$. Equation (11) follows immediately from (10) by noting:

$$
L_{(m,t)}^{(v,t+N(t))} < N(t)\delta, \text{ a.s.}, \forall m \in \mathcal{L}(t), v \in \mathcal{L}(t+N(t)). \quad \square
$$

B.4 Proof of Lemma 11

PROOF (LEMMA 11). The existence of an absorbing subset implies that the future sequence of descendant sets of (v,t) after time $t+m$ will simply be repeating copies of $\left\{ d_{(v,t)}(t+m+i) \right\}_{i=1}^{j_a K}$ and hence will never reach all of $\mathcal{L}(t')$ for any $t' > t$. The supremum in Equation (12) is hence unbounded and $\mathcal{L}$ is not synchronized by Theorem 9. $\square$

B.5 Proof of Proposition 12

PROOF (PROPOSITION 12). Suppose for some $j \geq 1$, neither 1) nor 2) has happened. Then the sequence $\{d_{(v,t)}(t+iK)\}_{i=1}^{j}$ must consist of unique subsets of $\mathcal{L}(t)$ (Recall that $\mathcal{L}(t) = \mathcal{L}(t+K)$, since K is the length of the scheduling frame). Because there are in total 2^n subsets for a set with cardinality n, in at most $2^{|\mathcal{L}(t)|}$ steps either of 1) or 2) will happen based on the Pigeonhole Principle. $\square$

B.6 Proof of Corollary 13

PROOF (COROLLARY 13). The corollary is a simple consequence of Theorem 7 and Lemma 18 (Section A of this Appendix). Given

$$\mathcal{L}(t') \text{ is totally connected with } \mathcal{L}(t'+jK),$$

Lemma 18 implies that

$$\mathcal{L}(t') \text{ is totally connected with } \mathcal{L}(t'+(j+1)K), \forall t' \in \mathbb{N}.$$

Therefore, for all $t \geq 1+(j+1)K$, the claim follows from Equation (11), by having $N(t) = (j+1)K$. $\square$

B.7 Proof of Proposition 17

PROOF (PROPOSITION 17). Consider the precedence graph V_P constructed from $\mathcal{L}$ and $\mathcal{G}_I$. It suffices to show that $\mathcal{L}(t)$ totally connects $\mathcal{L}(t+M-3)$ for all t. We can then invoke Corollary 13 to prove the upper-bound. Notice that compared to the upper-bound in Corollary 13, $j+1$ becomes j ($j=1$ in this case), because in this special case the minimum number of slots to achieve total connectedness starting from $\mathcal{L}(t)$ is the same for all $t \geq 1$. Let $T(v,t)$ be defined as:

$$T(v,t) := \inf \left\{ \delta : d_{(v,t)}(t+\delta) = \mathcal{L}(t+\delta), \delta \in \mathbb{N} \right\}.$$

From Figure 9, it is not difficult to check that

$$T(1,1) = \max_{v\in\mathcal{L}(t)} T(v,t), \forall t \geq 1.$$

Intuitively, this means it takes the largest number of steps for the descendant sets of the edges of the line to reach all vertices in some future $\mathcal{L}(t)$. Hence, it suffices to show that

$$T(1,1) = M - 3.$$

This is immediate by observing the following evolution of the sequence $\{d_{(1,1)}(t)\}_{t\geq 2}$, which is partially illustrated by the filled (gray) vertices in Figure 9:

$$d_{(1,1)}(1+3) = \{1,4\}, d_{(1,1)}(1+6) = \{1,4,7\},$$
$$\ldots$$
$$d_{(1,1)}(1+i\cdot 3) = \{1,4,7,\ldots,1+i\cdot 3\},$$
$$\ldots$$
$$d_{(1,1)}(1+M-3) = \{1,4,7,\ldots,M-2\} = \mathcal{L}(1+M-3),$$

which completes the proof. $\square$