Convergence and Stability of a Distributed CSMA Algorithm for Maximal Network Throughput

Libin Jiang and Jean Walrand
EECS Department, University of California at Berkeley
{ljiang,wlr}@eecs.berkeley.edu

***Abstract*—Designing efficient scheduling algorithms is an important problem in a general class of networks with resource-sharing constraints, such as wireless networks and stochastic processing networks. In [4], we proposed a distributed scheduling algorithm that can achieve the maximal throughput in such networks under certain conditions. This algorithm was inspired by CSMA (Carrier Sense Multiple Access). In this paper, we prove the convergence and stability of the algorithm, with properly-chosen step sizes and update intervals. Convergence of the joint scheduling and congestion control algorithm for utility maximization in [4] can be proved similarly.**

***Index Terms*—Distributed scheduling, maximal throughput, stochastic approximation, Markov process, convex optimization**

I. Introduction

Efficient resource allocation is essential to achieve high utilization of a class of networks with resource-sharing constraints, such as wireless networks and stochastic processing networks (SPN [6]). In wireless networks, certain links can not transmit at the same time due to the interference constraints among them. In a task processing problem (further explained later), two tasks can not be processed simultaneously if they both require monopolizing a common resource. A scheduling algorithm determines which link to activate (or which task to process) at a given time without violating these constraints. Designing distributed scheduling algorithms to achieve high throughput is an important problem [1], [11].

This paper is devoted to a proof of the convergence and stability of a simple-to-implement distributed scheduling algorithm for such networks proposed in [4], [5]. For ease of reference, we review the algorithm below. The algorithm avoids the need to search for a maximum weighted independent set as required by Maximal-Weight Scheduling [11], an algorithm that is known to be throughput-optimal but is not easy to implement, especially in a distributed way. (In [9], a similar randomized algorithm was independently proposed in the context of optical networks, and was later developed in [15].) The paper [4] also describes an algorithm that maximizes the utility of flows by combining scheduling and congestion control. The convergence of that algorithm can be proved using the same approach as in this paper.

Consider a wireless network where some links interfere. Packets arrive at the transmitters of the links with certain rates. Consider a "perfect CSMA" protocol [2], [3] that works as follows. The different transmitters choose independent exponentially-distributed backoff times. A transmitter decrements its backoff timer when it senses the channel idle and starts transmitting when its timer runs out. The packet transmission times are also exponentially distributed. (The process defines a "CSMA Markov chain".) The assumption in [2], [3] is that a transmitter hears any transmitter of a link that would interfere with it. That is, there are no hidden nodes. Moreover, the transmitters hear a conflicting transmission instantaneously. Accordingly, there are no collisions in perfect CSMA. In practice, other protocols such as RTS/CTS can be used to address the hidden node problems [2]. The optimality in the presence of collisions is analyzed in [12] (see also [14]). In the task processing problem, on the other hand, one can define a perfect CSMA protocol without considering collisions and hidden nodes.

The "adaptive CSMA" scheduling algorithm in [4] is as follows. Each link adjusts its transmission aggressiveness ("TA") based on its backlog. A link's TA is reflected in either its mean backoff time or mean transmission time. For example, the transmitter of a link sets its mean backoff time to be $\exp\{-\alpha \cdot Q\}$ where Q is the backlog of the link and $\alpha > 0$ is a small constant. That is, the link becomes more aggressive as its backlog increases. In [4], we have shown, under a time-scale-separation approximation, that such a simple algorithm is throughput-optimal (i.e., it stabilizes the queues if the arrival rates are strictly feasible). The approximation is that, as the links change their TA, the CSMA Markov chain instantaneously reaches its stationary distribution.

In this paper, we analyze the convergence and stability properties of the algorithm without the above approximation. In particular, we show that (i) For any strictly feasible arrival rates, using decreasing step sizes and increasing update intervals that satisfy certain conditions, the TA's of different links converge to the desired values. Although the intuition is to make the time-scale separation eventually hold, these conditions are quite intricate since the speed at which the CSMA Markov chain converges to its stationary distribution

Acknowledgement: Thanks to Prof. Devavrat Shah (MIT) who suggested using increasing lengths of update intervals in the algorithm, and Jinwoo Shin (MIT) for useful discussions. We also thank anonymous reviewers for their constructive comments. This research was supported in part by MURI grant BAA 07-036.18.

depends on the time-varying TA's. (ii) The maximal throughput can be arbitrarily approached by using constant step sizes and update intervals.

The rest of the paper is organized as follows. In section II, we describe the basic model and the throughput-optimality objective. Section III and IV present CSMA scheduling algorithms (adapted from [4], [5]), and give proofs of their convergence and/or stability under different sets of sufficient conditions. The same results apply to the joint algorithm in [4]. Section V provides simulation studies that illustrate the main results. We conclude the paper and discuss future research in section VI.

II. Basic Model and Problem statement

We first describe the basic model and objective as in [4].

A. Network Interference Model

There are K FIFO queues in the network. Not all queues can be served simultaneously, due to interference or resource-sharing constraints. These constraints are represented by a contention graph (or "CG") $G = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the set of vertexes (each of them represents a queue) and $\mathcal{E}$ is the set of edges. Two queues cannot be served at the same time (i.e., "conflict") if and only if there is an edge between them.

In wireless networks, one can associate a queue with each *link*, which is an ordered transmitter-receiver pair. Two links cannot be activated at the same time if they interfere. Although this is a simplified model for wireless networks, it does provide a useful abstraction and has been used widely in literature [2] [1].

In the task processing problem, assume K different types of tasks and a finite set of resources $\mathcal{B}$. A queue is associated with each type of tasks. To perform a type-k task, one needs a subset $\mathcal{B}_k \subseteq \mathcal{B}$ of resources and these resources are then monopolized by the task while it is being performed. Note that two tasks cannot be performed simultaneously iff they require some common resources. Clearly, this can be modeled by a conflict graph G defined above.

Assume that G has N different Independent Sets ("IS", not confined to "Maximal Independent Sets"), where each IS is a set of queues that can be served simultaneously. Denote the i'th IS as $x^i \in \{0,1\}^K$, a 0-1 vector that indicates which links are transmitting in this IS. That is, the k'th element of x^i, $x^i_k = 1$ if link k is transmitting, and $x^i_k = 0$ otherwise.

B. Throughput-optimality Objective

We now describe the *scheduling problem* which is the focus of the paper. Without loss of generality, assume that the capacity of each link is 1. Assume that traffic arrives at link k with an arrival rate $\lambda_k \in (0,1)$. For simplicity, assume the following i.i.d. Bernoulli arrivals (although it can be readily generalized [13]): Let $a_k(t) \in \{0,1\}$ be the arrival process at link k. For $t \in [j, j+1], j = 1, 2, \ldots$ (i.e., in a given "slot" with length 1), $a(t) = 1$ with probability λ_k and $a(t) = 0$ otherwise. Then, $A_k(t) := \int_0^t a_k(\tau)d\tau$, the cumulative amount of arrived traffic by time t, satisfies that $E(A_k(t))/t = \lambda_k$. Denote the vector of arrival rates as λ. We say that λ is *feasible* iff it can be written as $\lambda = \sum_i \bar{p}_i \cdot x^i$ where $\bar{p}_i \geq 0$ and $\sum_i \bar{p}_i = 1$. That is, there is a schedule of the independent sets (including the non-maximal ones) that can serve the arrivals. We say that λ is *strictly feasible* iff it can be written as $\lambda = \sum_i \bar{p}_i \cdot x^i$ where $\bar{p}_i > 0$ and $\sum_i \bar{p}_i = 1$. Denote the set of feasible and strictly feasible λ by $\bar{\mathcal{C}}$ and $\mathcal{C}$ respectively. It is not difficult to show that $\mathcal{C}$ is exactly the interior of $\bar{\mathcal{C}}$.

Our objective is to give a distributed scheduling algorithm such that any strictly feasible λ can be "supported". More formally, denote by $D_k(t)$ the cumulative traffic that has departed by t. The system is "rate stable" if $\lim_{t\to\infty}[A_k(t) - D_k(t)]/t = 0, \forall k$ almost surely. Another notion of stability is the positive (Harris) recurrence of the network Markov chain. An algorithm is said to be "throughput-optimal" if for any $\lambda \in \mathcal{C}$, it makes the system rate stable or positive (Harris) recurrent.

The convergence and stability results in this paper can be readily applied to the joint scheduling and congestion control algorithm proposed in [4].

III. A distributed CSMA Algorithm and its throughput-optimality

We first describe an idealized CSMA model proposed in [2], [3] which the algorithm in [4] in based on. Before transmitting, link k waits (or "backs-off") for a random period of time that is exponentially distributed with mean $1/R_k$. If it does not sense another transmission of a conflicting link during that time, then the link starts transmitting; otherwise, it suspends its backoff and resumes it after the conflicting transmission is over. The transmission time of link k is exponentially distributed with mean 1. Define $r_k = \log(R_k)$ as the "transmission aggressiveness" (TA) of link k. And let $\mathbf{r}$ be the vector of r_k's. Assuming that the sensing time is negligible, given the continuous distribution of the backoff times, collisions do not occur in the model of [2], [3].

Note that collision is not an issue in the task processing problem (cf. section II-A). In wireless networks, however, collisions occur since in practice the backoff time of each link is usually multiples of "minislots" due to the non-zero sensing time. Therefore the above idealized CSMA model provides an approximation. The approximation is more accurate when the transmission probability in each minislot is small which leads to small collision probability. In that case, the transmission time should be increased to compensate for the increases backoff time. In [12], we formulated a model which explicitly considers collisions among control packets such as RTS in 802.11, designed algorithms similar to those in [4], and provided their convergence and stability properties. (Both methods in this paper and in [12] can be used to provide convergence/stability results for another discrete time protocol [14] that considered collisions.) In this paper, we will focus on the case without collisions.

The transitions of the transmission states x^i form a Continuous Time Markov Chain, which is called the *CSMA Markov Chain*. References [2], [3] showed that the Markov chain (with a given $\mathbf{r}$) is *time-reversible*, and in the stationary distribution, the probability of state x^i is

$$p(x^i;\mathbf{r}) = \frac{\exp(\sum_{k=1}^K x_k^i r_k)}{C(\mathbf{r})} \quad (1)$$

where

$$C(\mathbf{r}) = \sum_j \exp(\sum_{k=1}^K x_k^j r_k) \ . \quad (2)$$

(Note that an IS with larger $\sum_{k=1}^K x_k^i r_k$ has a higher probability.) Then, the probability that link k is active is

$$s_k(\mathbf{r}) := \sum_i [x_k^i \cdot p(x^i;\mathbf{r})]. \quad (3)$$

Since the link capacity is assumed to be 1, $s_k(\mathbf{r})$ is also the average service rate (or throughput) of link k given $\mathbf{r}$.

For simplicity, we assume that the arrival traffic can be viewed as "fluid". That is, upon transmission, the packet sizes may be different from the sizes of the arrived packets (by re-packetize the bits in the queue). This assumption, however, is not essential. More discussion is given in [13].

A. Review of the ideas behind the Algorithms

The algorithms in [4], [5] try to find or approximate, in a distributed way, the TA vector $\mathbf{r}$ in CSMA such that the induced service rates (3) at all links are not less than the arrival rates λ whenever λ is strictly feasible. In this section, we review some results in [4], [5] which state that the desired $\mathbf{r}$ can be obtained as the optimal dual variables in some convex optimization problems ((6) or (7)).

Consider the following convex optimization problem, where $\mathbf{u} \in \mathcal{R}_+^N$ is a probability distribution over the IS's (recall that N is the number of IS's). Let $\mathcal{P} := \{\mathbf{u}' \in \mathcal{R}_+^N | \sum_i u'_i = 1\}$ be the set of such distributions.

$$\begin{array}{ll} \max_{\mathbf{u}\in\mathcal{P}} & -\sum_i u_i \log(u_i) \\ \text{s.t.} & \sum_i (u_i \cdot x_k^i) \geq \lambda_k, \forall k \end{array} \quad (4)$$

where λ is strictly feasible, and $\sum_i$ is the summation over all IS's.

Lemma 1: ([4]) For all k, let $r_k^* \geq 0$ be the (unique) optimal dual variable associated with the constraint $\sum_i (u_i \cdot x_k^i) \geq \lambda_k$ in (6). Then $\mathbf{r}^*$ satisfies that

$$s_k(\mathbf{r}^*) \geq \lambda_k, \forall k,$$

that is, with the TA vector $\mathbf{r}^*$, the service rate (3) at any link is high enough. (And the optimal $\mathbf{u}^*$ is the corresponding stationary distribution of the CSMA Markov chain.) Also, an iterative (subgradient dual) algorithm to find $\mathbf{r}^*$ (by solving the dual problem $\min_{\mathbf{r}\geq\mathbf{0}} L_1(\mathbf{r})$) is (for $j = 1, 2, \ldots$)

$$r_k(j) = [r_k(j-1) + \alpha(j)(\lambda_k - s_k(\mathbf{r}(j-1)))]_+, \forall k \quad (5)$$

where $\alpha(j)$ is some properly-chosen step size. *That is, link k increases r_k if the service rate is smaller than λ_k, and vice versa.*
The proof is given in [13].

Similarly, for the optimization problem

$$\begin{array}{ll} \max_{\mathbf{u}\in\mathcal{P}} & -\sum_i u_i \log(u_i) \\ \text{s.t.} & \sum_i (u_i \cdot x_k^i) = \lambda_k, \forall k, \end{array} \quad (6)$$

the (unique) vector of optimal dual variables $\mathbf{r}^*$ satisfies that $s_k(\mathbf{r}^*) = \lambda_k, \forall k$.

The next optimization problem is an extension of (6) such that the optimal dual variables $\mathbf{r}^*$ satisfies $s_k(\mathbf{r}^*) > \lambda_k, \forall k$. The strict inequality can be used later to ensure that the queue lengths are stable and tend to be small.

$$\begin{array}{ll} \max_{\mathbf{u}\in\mathcal{P},\mathbf{w}} & -\sum_i u_i \log(u_i) + c\sum_k \log(w_k) \\ \text{s.t.} & \sum_i (u_i \cdot x_k^i) \geq \lambda_k + w_k, \forall k \\ & 0 \leq w_k \leq \bar{w}, \forall k \end{array} \quad (7)$$

where λ is strictly feasible, and $c > 0, \bar{w} > 0$ are small constants.

Lemma 2: ([5]) For all k, let $r_k^* \geq 0$ be the (unique) optimal dual variable associated with the constraint $\sum_i (u_i \cdot x_k^i) \geq \lambda_k + w_k$ in problem (7). Then $\mathbf{r}^*$ satisfies that

$$s_k(\mathbf{r}^*) > \lambda_k, \forall k.$$

Also, a (subgradient dual) algorithm to find $\mathbf{r}^*$ (by solving the dual problem $\min_{\mathbf{r}\geq\mathbf{0}} L_2(\mathbf{r})$) is (for $j = 1, 2, \ldots$)

$$\begin{array}{rcl} r_k(j) & = & [r_k(j-1) + \alpha(j)(\lambda_k - s_k(\mathbf{r}(j-1)) + \\ & & \min\{c/r_k(j-1), \bar{w}\})]_+, \forall k \end{array} \quad (8)$$

where $\alpha(j)$ is some properly-chosen step size. (The proof is similar to that of Lemma 1, and is given in [13].)

However, algorithms (5) or (8) require the knowledge of λ_k and $s_k(\mathbf{r}(j-1))$, which cannot be obtained directly in the network since both the traffic arrival and service processes are random. Therefore in the actual algorithm we need to properly average the randomness. The main complication here is that the time needed for the CSMA Markov chain to converge to its stationary distribution (i.e., the mixing time) depends on the varying $\mathbf{r}$. So the dynamics of the Markov chain and $\mathbf{r}$ are coupled in a complex way. The goal here is to provide sufficient conditions to ensure the convergence of the algorithm with random arrivals and service.

B. TA adjustment Algorithm

Let $x_k(t) \in \{0, 1\}$ be the instantaneous state of link k at (continuous) time t. For link k, define the cumulative "service" by time t as $S_k(t) = \int_{\tau=0}^t x_k(\tau)d\tau$, and the cumulative departure by time t as $D_k(t) = \int_{\tau=0}^t x_k(\tau)I(Q_k(\tau) > 0)d\tau$, where $I(\cdot)$ is the indicator function and $Q_k(\tau) := Q_k(0) + A_k(\tau) - D_k(\tau)$ is the queue length of link k at time τ. Note that there is no departure if the queue is empty but we allow $x_k(\tau) = 1$ even if $Q_k(\tau) = 0$ (in which case dummy packets are sent, further discussed below).

The adaptive CSMA algorithm which adjusts the TA is given below (Notice its similarity to (8)). The algorithm is *fully distributed* and requires *no exchange of control messages*. We assume that there is a maximal instantaneous arrival rate $\bar{\lambda}$ for any link.

Algorithm 1: The vector $\mathbf{r}$ is updated at time t_f, $j = 1, 2, \ldots$. Let $t_0 = 0$ and $T_j := t_j - t_{j-1}, j = 1, 2, \ldots$. Define "period j" as the time between t_{j-1} and t_j, and $\mathbf{r}(j)$ be the value of $\mathbf{r}$ at the end of period j, i.e., at time t_j.

Initially, set $\mathbf{r}(0) = \mathbf{0}$. Then at time t_j ($j = 1, 2, \ldots$), update

$$r_k(j) = [r_k(j-1) + \alpha(j)(\lambda'_k(j) - s'_k(j) + \min\{c/r_k(j-1), \bar{w}\})]_+ \quad (9)$$

for all k, where $\lambda'_k(j)$ and $s'_k(j)$ are the empirical average arrival rate and service rate of link k in period j (i.e., $\lambda'_k(j) = [A_k(t_j) - A_k(t_{j-1})]/T_j \le \bar{\lambda}$, $s'_k(j) = [S_k(t_j) - S_k(t_{j-1})]/T_j$). $c > 0, \bar{w} > 0$ are small constants. We let link k transmit dummy packet with TA $r_k(j)$ even if the queue is empty. This ensures that the CSMA Markov chain (with parameter $\mathbf{r}(j)$) has the desired stationary distribution (1). (The transmitted dummy packets are included when computing $s'_k(j)$.)

Also, $\alpha(j)$ and T_j are chosen such that

$$\alpha(j) > 0, \sum_j \alpha(j) = \infty, \sum_j \alpha^2(j) < \infty \quad (10)$$

$$\sum_{m=0}^{\infty} [\alpha(m+1) \sum_{j=1}^{m} \alpha(j)]^2 < \infty \quad (11)$$

$$\sum_{m=0}^{\infty} [\alpha(m+1) \cdot (\sum_{j=1}^{m} \alpha(j)) \cdot f(m)/T_{m+1}] < \infty \quad (12)$$

where

$$f(m) := \exp\{(\tfrac{5}{2}K + 1) \cdot [\lambda_{max} \cdot \sum_{j=1}^{m} \alpha(j) + \log(2)]\} \quad (13)$$

where K is the number of links, and $\lambda_{max} = \bar{\lambda} + \bar{w}$.

Remark 1: According to (9), the algorithm does not need to know λ_k explicitly.

Remark 2: In an alternative design, the mean backoff time of each link is 1, and the mean transmission time of link k is $\exp(r_k)$. For a given $\mathbf{r}$, the CSMA Markov chain has the same stationary distribution as in (1). In that case, the same Algorithm 1 can be used, with a minor difference in the definition of (13). More details are given in [13].

Proposition 1: The setting $\alpha(j) = 1/[(j+1)\log(j+1)]$ and $T_j = j$ satisfies conditions (10), (11) and (12). Note that this setting *does not depend on, or require the knowledge of K and λ_{max}, and thus can generally apply to any network.*

Similarly, the same is true for the following settings. (i) $\alpha(j) = 1/[(j+1)\log(j+1)]$ and $T_j = j^\gamma$ for any $\gamma > 0$; (ii) $\alpha(j) = c_0/[(a \cdot j + b + 1)\log(a \cdot j + b + 1)]$ and $T_j = a \cdot j + b$ (with constants $a > 0, b > 0, c_0 > 0$).

The above is not difficult to check [13].

A main result of the paper is the following:

Theorem 1: Assume that λ is strictly feasible (i.e., $\lambda \in \mathcal{C}$). Then with Algorithm 1, $\mathbf{r}(j)$ converges to some $\mathbf{r}^*$ with probability 1. The vector $\mathbf{r}^*$ satisfies that $s_k(\mathbf{r}^*) > \lambda_k, \forall k$. Also, the system is rate stable.

The proof of Theorem 1 is in the Appendix and [13].

Remark: Another notion of stability often used in literature is the positive (Harris) recurrence of the underlying network Markov process, in particular the queue lengths. Note that with the time-varying step sizes and update intervals in Algorithm 1, the Markov process is not time-homogeneous, in which case positive (Harris) recurrence is not well defined. This is the reason why we choose to prove the "rate stability". One concern for rate stability is that the queue lengths may go to infinity. However, this does not happen in Algorithm 1 since $\mathbf{r}$ converges to $\mathbf{r}^*$ such that $s_k(\mathbf{r}^*) > \lambda_k, \forall k$. Using the fact that $\lim_{t\to\infty} S_k(t)/t = s_k(\mathbf{r}^*)$ (Lemma 4 in [13]), one can show that the queue lengths return to around 0 infinite times:

Proposition 2: Let $Q_k(t)$ be the queue size of link k at (continuous) time t. Consider the process $\{Q_k(t), t = 0, 1, 2, \ldots\}$. With Algorithm 1, for any link k, $Q_k(t) \le 2$ infinite times (w. p. 1) in the above process.

In a related work, reference [10] used a differential-equation method to analyze the convergence of the utility maximization algorithm extended from [4]. In [10], an upper bound of $\mathbf{r}^*$ needs to be known beforehand to bound $\mathbf{r}(j)$ in the algorithm. Therefore, it is not obvious whether the proof there can be directly applied to the scheduling problem above without a priori upper bound of $\mathbf{r}^*$. Also, the queue stability was not considered in [10].

C. An Algorithm with bounded TA (reduced capacity)

We have shown above that Algorithm 1 is throughput-optimal in that it can support any $\lambda \in \mathcal{C}$. No upper bound of TA is imposed in Algorithm 1. In this section, we give similar algorithms which simply upper-bound the TA by a constant $r_{max} > 0$. The algorithm's capacity region is smaller than $\mathcal{C}$. But it allows weaker conditions on the step sizes and update intervals. Also, one can choose the parameters of the algorithm to make its capacity region arbitrarily close to $\mathcal{C}$.

Algorithm 2: The vector $\mathbf{r}$ is updated at time t_j, $j = 1, 2, \ldots$.

$$r_k(j) = [r_k(j-1) + \alpha(j)(\lambda'_k(j) + \epsilon - s'_k(j))]_{[0, r_{max}]}, \forall k. \quad (14)$$

where $\epsilon > 0$, and $[\cdot]_{[0,r_{max}]}$ means the projection to the set $[0, r_{max}]$. Algorithm 2 tries to solve problem (6) (notice its similarity to (5)), except that it "pretends" to serve the arrival rates $\lambda + \epsilon \cdot \mathbf{1}$ which are higher than the actual arrival rates λ, in order to ensure that the average service rate is strictly higher than the arrival rate after convergence.

Also, $\alpha(j)$ and T_j are required to satisfy (10) and

$$\sum_{m=0}^{\infty} [\alpha(m+1)/T_{m+1}] < \infty \quad (15)$$

For example, $\alpha(j) = 1/j$ and $T_j = j^\gamma$ for any $\gamma > 0$ satisfy (10) and (15).

The following theorem states that the capacity region of Algorithm 2 is (at least)

$$\mathcal{C}_{II}(r_{max}, \epsilon) = \{\lambda | \lambda + \epsilon \cdot \mathbf{1} \in \mathcal{C} \text{ and } \mathbf{r}^*_{II}(\lambda + \epsilon \cdot \mathbf{1}) \in [0, r_{max}]^K\}$$

where $\mathbf{r}^*_{II}(\lambda + \epsilon \cdot \mathbf{1})$ is the optimal vector of dual variables $\mathbf{r}^*$ of problem (6) with arrival rates $\lambda + \epsilon \cdot \mathbf{1}$.

Theorem 2: With Algorithm 2, if $\lambda \in \mathcal{C}_{II}(r_{max}, \epsilon)$, then with probability 1, $\mathbf{r}(j)$ converges to $\mathbf{r}^*_{II}(\lambda + \epsilon \cdot \mathbf{1})$. (And $s_k(r^*_{II}(\lambda + \epsilon \cdot \mathbf{1})) \ge \lambda_k + \epsilon > \lambda_k, \forall k$.) So the queues are rate stable and return to around zero infinite times (similar to Prop. 2).

The proof (given in [13]) is a minor modification of the proof of Theorem 1 by utilizing the boundedness of $\mathbf{r}$.

Clearly, $\mathcal{C}_{II}(r_{max}, \epsilon) \to \mathcal{C}$ as $r_{max} \to +\infty$ and $\epsilon \to 0$. So we can choose r_{max}, ϵ to achieve arbitrarily close approximations of the maximal capacity region $\mathcal{C}$.

It can be also shown that one can use constant $T_j = T, \forall j$ and decreasing step sizes $\alpha(j)$ to achieve convergence and stability.

Algorithm 3: Use the updates $r_k(j) = [r_k(j-1) + \alpha(j)(\lambda'_k(j) + \epsilon - s'_k(j))]_{[r_{min}, r_{max}]}$, where $\alpha(j)$ is decreasing with j and satisfies (10). The update intervals $T_j = T, \forall j$ where $T > 0$ is any constant.

Theorem 3: Assume that $\lambda \in \mathcal{C}_{III}(r_{min}, r_{max}, \epsilon)$ where

$$\begin{aligned}\mathcal{C}_{III}(r_{min}, r_{max}, \epsilon) \quad = \quad & \{\lambda | \lambda + \epsilon \cdot \mathbf{1} \in \mathcal{C} \text{ and} \\ & \mathbf{r}^*(\lambda + \epsilon \cdot \mathbf{1}) \in (r_{min}, r_{max})^K\}\end{aligned}$$

where $\mathbf{r}^*(\lambda + \epsilon \cdot \mathbf{1})$ is the (unique) vector of optimal dual variables in problem (6) with the arrival rate $\lambda + \epsilon \cdot \mathbf{1}$. Then with Algorithm 3, $\mathbf{r}(j) \to \mathbf{r}^*(\lambda + \epsilon \cdot \mathbf{1})$, the queues are rate stable and return to around zero infinite times (similar to Prop. 2) with probability 1.

The proof involves dividing the time into "frames" where each frame consists of multiple update intervals. Then we bound the "error" of the changes of $\mathbf{r}$ in the frames (compared to the changes if we have the exact gradients.) The proof is omitted due to the space limit. (We note that the differential-equation method used in [10] can potentially provide another proof when $\mathbf{r}(j)$ is bounded, similar to the convergence result in the collision case [12].)

IV. Constant step size and update interval

Now we consider Algorithm 2 with a constant step size $\alpha(j) = \alpha, \forall j$ and a constant update interval $T_j = T, \forall j$. So unlike Algorithm 1, the network Markov process under Algorithm 2 is time-homogeneous.

Theorem 4: If $\lambda \in \mathcal{C}_{II}(r_{max}, \epsilon)$, then there exists $\alpha > 0, T > 0$ such that the queues are stable using Algorithm 2 with $\alpha(j) = \alpha, T_j = T, \forall j$.

The proof is given in [13].

The following bounds are useful to characterize the region $\mathcal{C}_{II}(r_{max}, \epsilon)$ and $\mathcal{C}_{III}(r_{min}, r_{max}, \epsilon)$.

Proposition 3: Given $\lambda \in C$. If $\delta_k > 0$ is the maximum value such that $\lambda + \delta_k \cdot \mathbf{e}_k \in \bar{C}$ (where $\mathbf{e}_k$ is the K-dimensional vector whose k'th element is 1 and others are 0's), then

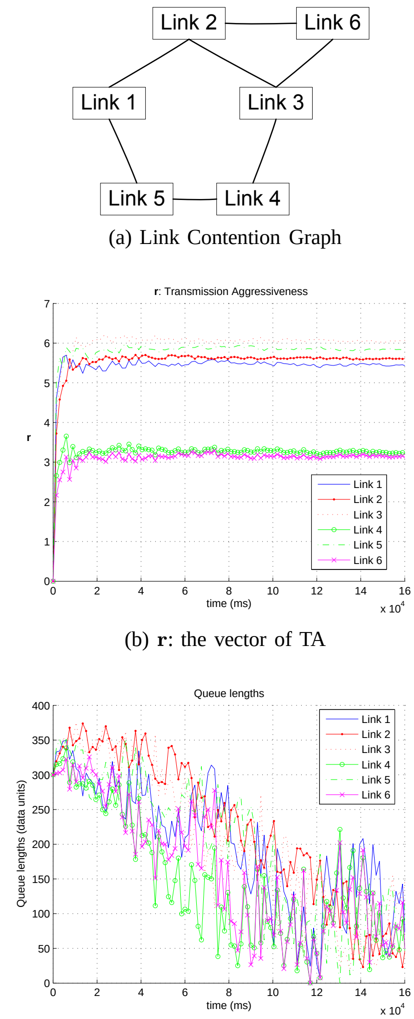
$$r^*_{II,k}(\lambda) \le \frac{\log(N)}{\delta_k}. \tag{16}$$

So, if $\delta_k \ge \log(N)/r_{max}, \forall k$, then $\lambda \in \mathcal{C}_{II}(r_{max}, 0)$. (A slightly looser bound was obtained in [16].)

We also have another bound which is tighter than (16), especially for small values of δ_k.

Proposition 4: Given $\lambda \in C$. Recall that $r^*_k(\lambda)$ denotes the unique $\mathbf{r}$ such that $s_k(\mathbf{r}) = \lambda_k, \forall k$. If $\delta_k > 0$ is the maximum value such that $\lambda + \delta_k \cdot \mathbf{e}_k \in \bar{C}$, then

$$\log(\frac{\lambda_k}{1-\lambda_k}) \le r^*_k(\lambda) \le b_k \cdot [\log(\frac{N}{b_k \cdot \delta_k}) + 1]$$



(a) Link Contention Graph

(b) $\mathbf{r}$: the vector of TA

(c) Queue lengths

Fig. 1: CSMA Scheduling with varying step sizes and update intervals

for some constant $b_k > 0$ which depend on the network. Note that as $\delta_k \to 0$, the upper bound is in the order of $\log(1/\delta_k)$, compared to $1/\delta_k$ in (16).

V. Numerical examples

In our C++ simulations, the transmission time of all links is exponentially distributed with mean 1ms, and the backoff time of link k is exponentially distributed with mean $1/\exp(r_k)$ ms. Assume that the capacity of each link is 1(data unit)/ms. The initial TA $r_k(0) = 0$ for all link k. To show the negative drift of queues, assume that initially, all queue lengths are 300 data units. Then r_k is adjusted using Algorithm 1, with step sizes $\alpha(j) = 0.46/[(2 + j/1000)\log(2 + j/1000)]$ and update intervals $T_j = (2 + j/1000)$ ms. The constants $c = 0.01$, and $\bar{w} = 0.02$.

There are 6 links in the network, whose conflict graph is shown in Fig. 1 (a). (Each link only needs to know the set of links that conflict with itself.) Define $0 \le \rho < 1$ as the "load factor", and let $\rho = 0.98$ in this simulation.

The arrival rate vector is set to λ=ρ*[0.2*(1,0,1,0,0,0) + 0.3*(1,0,0,1,0,1) + 0.2*(0,1,0,0,1,0) + 0.3*(0,0,1,0,1,0)] = ρ*(0.5,0.2,0.5,0.3,0.5,0.3) (data units/ms). We have multiplied by ρ a convex combination of some maximal IS's to ensure that λ is in the interior of the capacity region.

As expected, the TA vector $\mathbf{r}$ tends to converge (Fig. 1 (b)). Also, the queues tend to decrease and are stable (Fig. 1 (c)).

Due to the limit of space, simulation results with constant step size are given in [13].

VI. Conclusion

This paper has provided proofs of the convergence and stability property of the distributed CSMA scheduling algorithm proposed in [4], [5] with properly chosen step sizes and update intervals. Similar results also apply to the cross-layer algorithm (joint CSMA scheduling and congestion control) in [4], [5].

The conditions on the step sizes and update intervals given here are sufficient for the convergence/stability of the algorithms. However, since certain bounds in the proof may not be tight, it is possible that these conditions are not necessary. Also, we have assumed general conflict graphs. In many networks of practical interest, however, the conflict graphs may have particular structures. For example, if the conflict graph is a full graph (corresponding to a network where all links conflict to each other), then it can be shown that the mixing time is much smaller than the worst-case bound used in this paper. In the future, we would like to study whether some of the conditions can be relaxed, either generally or in networks with certain structure.

VII. Appendix

A. Some notation

Before proving Theorem 1, we need some further notation. Let $x^0(m-1)$ be the state of the CSMA Markov chain at the beginning of period m (i.e., at time t_{m-1}). Define the random vector $U(m-1) := (\mathbf{s}'(m-1), \lambda'(m-1), \mathbf{r}(m-1), x^0(m-1))$ for $m > 1$ and $U(0) = (\mathbf{r}(0) = \mathbf{0}, x^0(0))$. For $m \geq 1$, let $\mathcal{F}_{m-1}$ be the σ-field generated by $U(0), U(1), \ldots, U(m-1)$.

Given a vector of TA $\mathbf{r}(m-1)$ at the beginning of the period m of Algorithm 1, the vector $\mathbf{g}(m)$ whose k-th element $g_k(m) := s_k(\mathbf{r}(m-1)) - \lambda_k - (c/r_k(m-1)) \wedge \bar{w}$ is a subgradient of $L_2(\mathbf{r})$ (the dual problem of (7) is $\min_{\mathbf{r}\geq\mathbf{0}} L_2(\mathbf{r})$: see the proof of Lemma 2 in [13]). To find the desired $\mathbf{r}^*$ which solves the dual problem, the ideal algorithm (8) would follow the opposite direction of $\mathbf{g}(m)$. However, Algorithm 1 only has an estimation of $g_k(m)$, denoted by

$$g'_k(m) = s'_k(m) - \lambda'_k(m) - (c/r_k(m-1)) \wedge \bar{w}. \quad (17)$$

The "error bias" of $g'_k(m)$ is defined as

$$\begin{aligned} B_k(m) : &= E[g'_k(m)|\mathcal{F}_{m-1}] - g_k(m) \\ &= E[s'_k(m)|\mathcal{F}_{m-1}] - s_k(\mathbf{r}(m-1)) - \\ &\quad [E[\lambda'_k(m)|\mathcal{F}_{m-1}] - \lambda_k]. \end{aligned} \quad (18)$$

Define also the zero-mean "noise"

$$\begin{aligned} \eta_k(m) : &= (s'_k(m) - E[s'_k(m)|\mathcal{F}_{m-1}]) \\ &\quad -(\lambda'_k(m) - E[\lambda'_k(m)|\mathcal{F}_{m-1}]). \end{aligned}$$

Since both $s'_k(m)$ and $\lambda'_k(m)$ are bounded, the noise is also bounded: $|\eta_k(m)| \leq c_2$ for some $c_2 > 0$. Then, we have

$$g'_k(m) = g_k(m) + B_k(m) + \eta_k(m). \quad (19)$$

B. Proof of Theorem 1

The proof is composed of two parts. The first part analyzes the mixing time of the CSMA Markov chain, and shows that with Algorithm 1 and condition (12), the error bias $\mathbf{B}(m)$ (18) decreases "fast enough" with time. The second part (Lemma 3) is related to the theory of *stochastic approximation* and proves the convergence of $\mathbf{r}(j)$ to $\mathbf{r}^*$, the optimal dual variables of problem (7). Essentially, part 1 has ensured that the bias $\mathbf{B}(m)$ diminishes as $m \to \infty$, and (11) ensures that the effect of the martingale noise $\eta(m)$ diminishes (by the martingale convergence theorem.) Combining these and the choice of step sizes, the convergence to $\mathbf{r}^*$ can be established.

In the following consider period $m+1$ (i.e., from t_m to t_{m+1}). At time t_m with the TA vector $\mathbf{r}(m)$, denote the corresponding CSMA Markov chain by $X(t)$ (for convenience we drop the index $m+1$). $X(t)$ is a continuous time Markov chain (CTMC). By (1), the probability of state $x \in \{0,1\}^K$ in the stationary distribution of $X(t)$ is

$$\pi_x(\mathbf{r}(m)) = p(x; \mathbf{r}(m)) = \frac{1}{C(\mathbf{r}(m))} \exp(\sum_k x_k r_k(m)).$$

Since $\mathbf{r}(m) \geq \mathbf{0}$, using (2),

$$C(\mathbf{r}(m)) \leq \sum_{x'} \exp(\mathbf{1}^T \mathbf{r}(m)) \leq 2^K \exp(\mathbf{1}^T \mathbf{r}(m))$$

since there are at most 2^K states. Also, $\exp(\sum_k x_k r_k(m)) \geq 1$ (since $r_k(m) \geq 0$ in Algorithm 1). So, the minimal probability in the stationary distribution

$$\pi_{min}(\mathbf{r}(m)) := \min_x \pi_x(\mathbf{r}(m)) \geq \exp(-\mathbf{1}^T \mathbf{r}(m) - K \cdot \log(2)).$$

Since $\lambda'_k(j) + \min\{c/r_k(j), \bar{w}\} \leq \lambda_{max}$ and $s'_k(\mathbf{r}(j)) \geq 0$, we have $r_k(j+1) \leq r_k(j) + \alpha(j)\lambda_{max}, \forall i, k$. Recall that $r_k(0) = 0, \forall k$. So $r_k(m) \leq \lambda_{max} \sum_{j=1}^m \alpha(j), \forall k$. Thus,

$$\pi_{min}(\mathbf{r}(m)) \geq \exp\{-K \cdot [\lambda_{max} \sum_{j=1}^m \alpha(j) + \log(2)]\}. \quad (20)$$

To proceed with the proof, we first "uniformize" $X(t)$. Recall that for the Markov chain $X(t)$, if each element of its transition rate matrix Q has an absolute value less than a constant A_{m+1}, then we can write $X(t) = Z(N(t))$ where $Z(n)$ is a discrete time Markov chain with probability transition matrix $P = I + Q/A_{m+1}$, where I is the identity matrix, and $N(t)$ is an independent Poisson process with rate A_{m+1}. We claim that $A_{m+1} = K \cdot \exp(\lambda_{max} \sum_{j=1}^m \alpha(j))$ suffices for the need. (Proof: $\because$ $r_k(m) \leq \lambda_{max} \sum_{j=1}^m \alpha(j)$, we have $Q(x,x') \le \exp(\lambda_{max}\sum_{j=1}^{m}\alpha(j)), \forall x, x'$. Also, for any state x, $Q(x,x')>0$ for at most K different x', i.e., state x can at most transit to K other states by changing the state of any one of the K links, so $\sum_{x'\neq x} Q(x,x') \le A_{m+1}$.)

Now we estimate how far $E[s'_k(m+1)|\mathcal{F}_m]$ is from the desired value $s_k(\mathbf{r}(m))$. Let the vector $\mu_m(t)=\{\mu_m(t,x)\}$ be the probabilities of all states at time t_m+t (where $0\le t \le T_{m+1}$), given that the initial state at time t_m is $x^0(m)$ and that the TA's during $[t_m, t_{m+1})$ are $\mathbf{r}(m)$. Let $x(t_m+t)=\{x_k(t_m+t)\}$ be the state at time t_m+t. Then

$$
\begin{aligned}
&E[s'_k(m+1)|\mathcal{F}_m]\\
&\quad= E[(1/T_{m+1})\cdot \int_0^{T_{m+1}} I(x_k(t_m+t)=1)dt]\\
&\quad= (1/T_{m+1})\cdot \int_0^{T_{m+1}} P(x_k(t_m+t)=1)dt\\
&\quad= (1/T_{m+1})\cdot \sum_{x':x'_k=1}[\int_0^{T_{m+1}} \mu_m(t,x')dt]\\
&\quad= \sum_{x':x'_k=1}\bar{\mu}_m(x')
\end{aligned}
$$

where $\bar{\mu}_m(x')=(1/T_{m+1})\cdot\int_0^{T_{m+1}}\mu_m(t,x')dt$ is the time-averaged probability of state x' in the interval. Let $\bar{\mu}_m := \{\bar{\mu}_m(x)\}$ be the vector of such probabilities of all states.

Let $\pi_{x^0}(\mathbf{r}(m))$ be the probability of $x^0(m)$, simply written as x^0, in the stationary distribution of $X(t)$. Use $||\cdot||_{var}$ to denote the variation distance between two distributions (expressed below). Let β_1 be the second largest eigenvalue of P, and the vector $\pi(\mathbf{r}(m)) := \{\pi_x(\mathbf{r}(m))\}$. The following inequality (a slight extension of Eq. (1.10) in [7]) has used the fact that $X(t)$ is equivalent to $Z(N(t))$,

$$
\begin{aligned}
||\mu_m(t)-\pi(\mathbf{r}(m))||_{var} &:= \sum_x |\mu_m(t,x)-\pi_x(\mathbf{r}(m))|/2\\
&\le \frac{1}{2}\sqrt{\frac{1-\pi_{x^0}(\mathbf{r}(m))}{\pi_{x^0}(\mathbf{r}(m))}}\exp(-A_{m+1}(1-\beta_1)t)\\
&\le \frac{1}{2}\sqrt{\frac{1}{\pi_{min}(\mathbf{r}(m))}}\exp(-A_{m+1}(1-\beta_1)t).
\end{aligned}
$$

So,

$$
\begin{aligned}
&||\bar{\mu}_m-\pi(\mathbf{r}(m))||_{var}\\
&\quad= ||(1/T_{m+1})\cdot\int_0^{T_{m+1}}[\mu_m(t)-\pi(\mathbf{r}(m))]dt||_{var}\\
&\quad\le (1/T_{m+1})\cdot\int_0^{T_{m+1}}||\mu_m(t)-\pi(\mathbf{r}(m))||_{var}dt\\
&\quad\le \frac{1}{2}\sqrt{\frac{1}{\pi_{min}(\mathbf{r}(m))}}\frac{1}{A_{m+1}(1-\beta_1)T_{m+1}} \qquad (21)
\end{aligned}
$$

where the first inequality has used the fact that $||\cdot||_{var}$ is a convex function.

We remark here that if we use the average service rate of the last $\tilde{T}_{m+1} := \min\{\hat{T}, T_{m+1}\}$ time unit (where $\hat{T}>0$) in each period $m+1$ to estimate $s_k(\mathbf{r}(m))$ (i.e., letting $s'_k(m+1)=[S_k(t_{m+1})-S_k(t_{m+1}-\tilde{T}_{m+1})]/\tilde{T}_{m+1}$), the bound (21) would be better, which can lead to weaker conditions on the step sizes and update intervals. The reason is that at the end of period $m+1$, $\mu_m(t)$ is closer to the stationary distribution $\pi(\mathbf{r}(m))$.

Continuing the proof, β_1 can be bounded by Cheeger's inequality [7]

$$\beta_1 \le 1-\phi^2/2 \qquad (22)$$

where ϕ is the "conductance" of P, defined as

$$\phi := \min_{S\subset\Omega, \pi(S)\in(0,1/2]}\frac{F(S,S^c)}{\pi_S(\mathbf{r}(m))}$$

where $\pi_S(\mathbf{r}(m)) = \sum_{x\in S}\pi_x(\mathbf{r}(m))$, and $F(S,S^c)$ is the "ergodic flow" from S to S^c: $F(S,S^c)=\sum_{x\in S, x'\in S^c}F(x,x') = \sum_{x\in S,x'\in S^c}\pi_x(\mathbf{r}(m))P(x,x')$. Then similar to [9], we have

$$
\begin{aligned}
\phi &\ge \min_{S\subset\Omega,\pi(S)\in(0,1/2]} F(S,S^c)\\
&\ge \min_{x\neq x', P(x,x')>0} F(x,x')\\
&= \min_{x\neq x', P(x,x')>0}\{\pi_x(\mathbf{r}(m))\cdot P(x,x')\}.
\end{aligned}
$$

For any $x\neq x'$ such that $P(x,x')>0$, it must be that $Q(x,x')>0$. Note that $Q(x,x')=1$ or $Q(x,x')=\exp(r_k(m))$ for some k, so $Q(x,x')\ge 1$. Hence, $P(x,x')=Q(x,x')/A_{m+1}\ge 1/A_{m+1}$. Combined with the last inequality, we find

$$\phi \ge \min_x \pi_x(\mathbf{r}(m))/A_{m+1} = \pi_{min}(\mathbf{r}(m))/A_{m+1}.$$

Using (22), $\beta_1 \le 1-[\pi_{min}(\mathbf{r}(m))/A_{m+1}]^2/2$. Thus $1/(1-\beta_1)\le 2\cdot A_{m+1}^2[\pi_{min}(\mathbf{r}(m))]^{-2}$. Plugging this into (21) and use (20), we have

$$
\begin{aligned}
&||\bar{\mu}_m-\pi(\mathbf{r}(m))||_{var}\\
&\quad\le A_{m+1}[\pi_{min}(\mathbf{r}(m))]^{-5/2}/T_{m+1}.\\
&\quad\le K\cdot f(m)/T_{m+1}
\end{aligned}
$$

where $f(m)$ is defined in (13). So,

$$
\begin{aligned}
&|E[s'_k(m+1)|\mathcal{F}_m]-s_k(\mathbf{r}(m))|\\
&\quad= |\sum_{x':x'_k=1}\bar{\mu}_m(x')-s_k(\mathbf{r}(m))|\\
&\quad\le 2||\bar{\mu}_m-\pi(\mathbf{r}(m))||_{var}\\
&\quad\le 2\cdot K\cdot f(m)/T_{m+1}, \forall k. \qquad (23)
\end{aligned}
$$

Also, with the Bernoulli arrival process $a_k(t)$ assumed in section II-B, it is easy to show that

$$|E[\lambda'_k(m+1)|\mathcal{F}_m]-\lambda_k| \le 1/T_{m+1}. \qquad (24)$$

Therefore, the error bias $B_k(m+1)$, defined in (18), satisfies $|B_k(m+1)| \le 2K\cdot f(m)/T_{m+1} + 1/T_{m+1} \le 3K\cdot f(m)/T_{m+1}$. Denote by $\mathbf{B}(m)$ the vector of $B_k(m)$'s. Since $|r_k(m)-r_k^*| \le \bar{r}+\lambda_{max}\sum_{i=1}^{m}\alpha(i)$, where $\bar{r}=\max_k r_k^*$, we show that the term $(\mathbf{r}(m)-\mathbf{r}^*)^T \cdot \mathbf{B}(m+1)$ is diminishing:

$$\begin{aligned} & \sum_{m=0}^{\infty} \alpha(m+1)|(\mathbf{r}(m)-\mathbf{r}^*)^T \cdot \mathbf{B}(m+1)| \\ \leq \quad & 3K^2 \sum_{m=0}^{\infty} \{\alpha(m+1)[\bar{r}+\lambda_{max} \sum_{i=1}^{m} \alpha(i)] \cdot f(m)/T_{m+1}]\} \\ < \quad & \infty \end{aligned} \tag{25}$$

where the last step is obtained using condition (12).

Lemma 3: If (25) and (11) hold, then with Algorithm 1, $\mathbf{r}$ converges to $\mathbf{r}^*$ (the optimal dual variables of problem (7)) with probability 1.

As noted before, the proof of Lemma 3 uses stochastic approximation, and is similar to that of Theorem 3.1 in [8], but with more intricacies. The complete proof is given in [13].

To conclude, with Algorithm 1, $\mathbf{r}$ converges to $\mathbf{r}^*$ such that $s_k(\mathbf{r}^*) > \lambda_k, \forall k$. Then the rate stability is proved in [13].

REFERENCES

[1] X. Lin, N.B. Shroff, R. Srikant, "A Tutorial on Cross-Layer Optimization in Wireless Networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1452-1463, Aug. 2006.

[2] X. Wang and K. Kar, "Throughput Modelling and Fairness Issues in CSMA/CA Based Ad-Hoc Networks," *IEEE Infocom 2005*, Miami, Mar. 2005.

[3] S. C. Liew, C. Kai, J. Leung, B. Wong, "Back-of-the-Envelope Computation of Throughput Distributions in CSMA Wireless Networks," http://arxiv.org//pdf/0712.1854

[4] L. Jiang and J. Walrand, "A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks," *the Forty-Sixth Annual Allerton Conference on Communication, Control, and Computing*, Sep. 23-26, 2008.

[5] L. Jiang and J. Walrand, "A Distributed Algorithm for Maximal Throughput and Optimal Fairness in Wireless Networks with a General Interference Model", Technical Report, UC Berkeley, Apr. 2008. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-38.html

[6] J. M. Harrison and R. J. Williams, "Workload interpretation for Brownian models of stochastic processing networks," *Mathematics of Operations Research*, 32 (2007), pp. 808-820.

[7] P. Diaconis and D. Strook, "Geometric bounds for eigenvalues of Markov chains," *Annals of Applied Probability*, 1 (1991), pp. 36-61.

[8] J. Zhang, D. Zheng, and M. Chiang, "The impact of stochastic noisy feedback on distributed network utility maximization," *IEEE Transactions on Information Theory*, vol. 54, no. 2, pp. 645-665, Feb. 2008.

[9] S. Rajagopalan and D. Shah, "Distributed Algorithm and Reversible Network", *Conference on Information Sciences and Systems*, Princeton, NJ, USA, Mar. 2008.

[10] J. Liu, Y. Yi, A. Proutiere, M. Chiang, and H. V. Poor, "Convergence and Tradeoff of Utility-Optimal CSMA," http://arxiv.org/abs/0902.1996.

[11] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, 36:1936-1948, Dec. 1992.

[12] L. Jiang and J. Walrand, "Approaching Throughput-Optimality in a Distributed CSMA Algorithm: Collisions and Stability," (invited), *ACM Mobihoc'09 S3 Workshop*, May 2009.

[13] L. Jiang, J. Walrand, "Convergence and Stability of a Distributed CSMA Algorithm for Maximal Network Throughput," Technical Report, UC Berkeley, Mar. 2009. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-43.html

[14] J. Ni and R. Srikant, "Distributed CSMA/CA algorithms for achieving maximum throughput in wireless networks," *Information Theory and Applications Workshop*, Feb. 2009.

[15] S. Rajagopalan, D. Shah, J. Shin, "Network adiabatic theorem: an efficient randomized protocol for contention resolution," *ACM Sigmetrics*, Jun. 2009.

[16] L. Jiang, D. Shah, J. Shin, J. Walrand, "Distributed Random Access Algorithm: Scheduling and Congesion Control," http://arxiv.org/abs/0907.1266