A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks

Libin Jiang
Jean Walrand

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2009-124
http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-124.html

August 23, 2009

Acknowledgement

This work is supported by MURI grant BAA 07-036.18.

A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks

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

***Abstract*—In multi-hop wireless networks, designing distributed scheduling algorithms to achieve the maximal throughput is a challenging problem because of the complex interference constraints among different links. Traditional maximal-weight scheduling (MWS), although throughput-optimal, is difficult to implement in distributed networks. On the other hand, a distributed greedy protocol similar to IEEE 802.11 does not guarantee the maximal throughput. In this paper, we introduce an adaptive CSMA scheduling algorithm that can achieve the maximal throughput distributively. Some of the major advantages of the algorithm are that it applies to a very general interference model and that it is simple, distributed and asynchronous. Furthermore, the algorithm is combined with end-to-end flow control to achieve the optimal utility and fairness of competing flows. Simulations verify the effectiveness of the algorithm. Also, the adaptive CSMA scheduling is a modular MAC-layer algorithm that can be combined with various protocols in the transport layer and network layer. Finally, the paper explores some implementation issues in the setting of 802.11 networks.**

***Index Terms*—Cross-layer optimization, joint scheduling and congestion control, maximal throughput, CSMA**

I. Introduction

In multi-hop wireless networks, it is important to efficiently utilize the network resources and provide fairness to competing data flows. These objectives require the cooperation of different network layers. The transport layer needs to inject the right amount of traffic into the network based on the congestion level and the MAC layer needs to serve the traffic efficiently to achieve high throughput. Through a utility optimization framework [1], this problem can be naturally decomposed into rate control at the transport layer and scheduling at the MAC layer.

It turns out that MAC-layer scheduling is the bottleneck of the algorithm [1]. In particular, it is not easy to achieve the maximal throughput through distributed scheduling, which in turn prevents full utilization of the wireless network. Scheduling is challenging since the conflicting relationships between different links can be complicated.

It is well known that maximal-weight scheduling (MWS) [18] is *throughput-optimal*. That is, that scheduling can support any incoming rates within the capacity region. In MWS, time is assumed to be slotted. In each slot, a set of non-conflicting links (called an "Independent Set", or "IS") that have the maximal weight are scheduled, where the "weight" of a set of links is the summation of their queue lengths. (This algorithm has also been applied to achieve 100% throughput in input-queued switches [19].) However, finding such a maximal-weighted IS is NP-complete in general and is hard even for centralized algorithms. So its distributed implementation is not trivial in wireless networks.

A few recent works proposed throughput-optimal algorithms for certain interference models. For example, Eryilmaz et al. [4] proposed a polynomial-complexity algorithm for the "two-hop interference model"[1]. Modiano et al. [5] introduced a gossip algorithm for the "node-exclusive model"[2]. The extensions to more general interference models, as discussed in [4] and [5], usually involves extra challenges. Sanghavi et al. [6] introduced an algorithm that can approach the throughput capacity (with increasing overhead) for the node-exclusive model.

On the other hand, by using a distributed greedy protocol similar to IEEE 802.11, reference [9] shows that only a fraction of the throughput region can be achieved (after ignoring collisions). The size of the fraction depends on the network topology and interference relationships. Reference [10] studied the impact of such imperfect scheduling on utility maximization in wireless networks. In [12], Proutiere et al. developed asynchronous random-access-based scheduling algorithms whose throughput performance is no less than some maximal scheduling algorithms, e.g. Maximum Size scheduling algorithms.

Our first contribution in this paper is to introduce a *distributed* adaptive CSMA (Carrier Sensing Multiple Access) algorithm for a general interference model. It is inspired by CSMA but may be applied to more general resource sharing problems (i.e., not limited to wireless networks). We show that if packet collisions are ignored (as in some of the above references), the algorithm can achieve maximal throughput. The optimality in the presence of collisions is studied in [26], [27]. The algorithm may not be directly comparable to the throughput-optimal algorithms mentioned above since it utilizes the carrier-sensing capability. But it does have a few distinct features:

- Each node only uses its local information (e.g., its back-

This work is supported by MURI grant BAA 07-036.18. Preliminary versions of the paper appeared as [2] [3].

[1]In this model, a transmission over a link from node m to node n is successful iff none the one-hop neighbors of m and n is in any conversation at the time.

[2]In this model, a transmission over a link from node m to node n is successful iff neither m nor n is in another conversation at the time.

log). No explicit control messages are required among the nodes.

- It is based on CSMA random access, which is similar to the IEEE 802.11 protocol and is easy to implement.
- Time is not divided into synchronous slots. Thus no synchronization of transmissions is needed.

In a related work, Marbach et al. [11] studied a model of CSMA with collisions. It was shown that under the "node-exclusive" interference model, CSMA can be made asymptotically throughput-optimal in the limiting regime of large networks with a small sensing delay. In [13], Rajagopalan and Shah independently proposed a throughput-optimal algorithm similar to ours in the context of optical networks. However, there are some notable differences (e.g., the use of Proposition 1 here). Also, utility maximization (discussed below) was not considered in [13].

Our second contribution is to combine the proposed scheduling algorithm with end-to-end flow control using a novel technique, to achieve fairness among competing flows as well as maximal throughput (sections III, IV). The performance is evaluated by simulations (section VI). We show that the proposed CSMA scheduling is a modular MAC-layer algorithm and demonstrate its combination with optimal routing, anycast and multicast (Appendix F). Finally, we considered some practical issues (e.g., packet collisions) in the setting of 802.11 networks (section VII).

There is extensive research in joint MAC and transport-layer optimization, for example [7] and [8]. Their studies have assumed the slotted-Aloha random access protocol in the MAC layer, instead of the CSMA-like protocol we consider here. Slotted-Aloha does not need to consume power in carrier sensing. On the other hand, CSMA is known to have a larger capacity region. (In this paper, we are primarily interested in the throughput performance.) Other related works assume physical-layer models which are quite different from ours. For example, [14] considered CDMA interference model; and [15] focused on time-varying wireless channel.

II. Adaptive CSMA for Maximal Throughput

A. Interference model

First we describe the general interference model we will consider in this paper. Assume there are K *links* in the network, where each *link* is an (ordered) transmitter-receiver pair. The network is associated with a link contention graph (or "LCG") $G = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the set of vertexes (each of them represents a link) and $\mathcal{E}$ is the set of edges. Two links cannot transmit at the same time (i.e., "conflict") iff there is an edge between them. Note that this framework includes the "node-exclusive model" and "two-hop interference model" mentioned above as two special cases.

Assume that G has N different Independent Sets ("IS", not confined to "Maximal Independent Sets"). 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. The k'th element of x^i, $x^i_k = 1$ if link k is transmitting, and $x^i_k = 0$ otherwise. We also refer to x^i as a "transmission state", and x^i_k as the "transmission state of link k".

B. An idealized CSMA protocol and the average throughput

We use an idealized model of CSMA as in [21], [22], [23]. This model makes two simplifying assumptions. First, it assumes that if two links conflict – because their simultaneous transmissions would result in incorrectly received packets – then either of the two links hears when the other one transmits. Second, the model assumes that this sensing is instantaneous. The first assumptions implies that there are no hidden nodes (HN). This is possible if the range of carrier-sensing is large enough [25].[3]) The second assumption is violated in actual systems because of the finite speed of light and of the time needed to detect a received power.

There are a few reasons for using this model in our context, although it makes some simplifying assumptions about collisions and the HN problem: (1) It is simple, tractable, and captures the essence of CSMA/CA; (2) Even without considering collisions and hidden nodes, distributed scheduling to achieve maximal throughput is not an easy problem, as discussed in the Introduction section. In most of the paper, we focus on the scheduling problem, without mixing it with the other issues. Similar approaches have been taken in related works, for example [9], [1]; (3) The scheduling algorithm we propose here is inspired by CSMA, but it may be applied to more general resource sharing problems[4] (i.e., not limited to wireless networks).

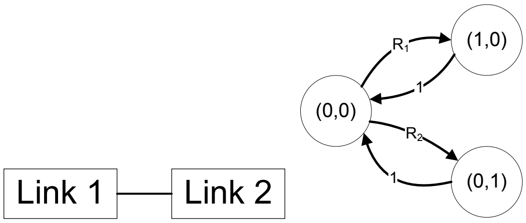
In [27], on the other hand, we have also developed a model that explicitly considers collisions in wireless network without HN. The distributed scheduling and rate control algorithms proposed in this paper can be naturally extended to that model. We will further discuss the issue in section VII.

In this subsection, assume that the links are always backlogged. If the transmitter of link k senses the transmission of any conflicting link (i.e., any link m such that $(k,m) \in \mathcal{E}$), then it keeps silent. If none of its conflicting links is transmitting, then the transmitter of link k waits (or backs-off) for a random period of time that is exponentially distributed with mean $1/R_k$ and then starts its transmission[5]. If some conflicting link starts transmitting during the backoff, then link k suspends its backoff and resumes it after the conflicting transmission is over. The transmission time of link k is exponentially distributed with mean 1. (The assumption on exponential distribution can be relaxed [23].) Assuming that the sensing time is negligible, given the continuous distribution of the backoff times, the probability for two conflicting links

[3] A related problem that affects the performance of wireless networks is the exposed-node (EN) problem. Reference [25] proposed a protocol to address HN and EN problems in a systematic way. We assume in this paper that the HN and EN are negligible with the use of such a protocol. Note that however, although EN problem may reduce the capacity region, it does not affect the applicability of our model, since we can define an edge between two links in the LCG as long as they can sense the transmission of each other, even if this results in EN.

[4] An example is the "task processing" problem described as follows. There are K different types of tasks and a finite set of resources $\mathcal{B}$. 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 can be performed simultaneously iff they use disjoint subsets of resources. Clearly this can be accommodated in our model in section II-A by associating each type of tasks to a "link".

[5] If more than one backlogged links share the same transmitter, the transmitter maintains independent backoff timers for these links.



(a) Link contention graph (b) CSMA Markov Chain

Fig. 1: Example: link contention graph and corresponding Markov Chain.

to start transmission at the same time is zero. So in the model of [21], [22], [23], collisions are ignored. (In section VII, however, we will discuss adaptations of our algorithm which consider collisions in an 802.11 network.)

It is not difficult to see that the transitions of the transmission states form a Continuous Time Markov Chain, which is called the *CSMA Markov Chain*. Denote link k's neighboring set by $\mathcal{N}(k) := \{m : (k,m) \in \mathcal{E}\}$. If in state x^i, link k is not active ($x^i_k = 0$) and all of its conflicting links are not active (i.e., $x^i_m = 0, \forall m \in \mathcal{N}(k)$), then state x^i transits to state $x^i + \mathbf{e}_k$ with rate R_k, where $\mathbf{e}_k$ is the K-dimension vector whose k'th element is 1 and all other elements are 0's. Similarly, state $x^i + \mathbf{e}_k$ transits to state x^i with rate 1. However, if in state x^i, any link in its neighboring set $\mathcal{N}(k)$ is active, then state $x^i + \mathbf{e}_k$ does not exist.

Fig 1 gives an example network whose LCG is shown in (a). There are two links, with an edge between them, which means that they cannot transmit together. Fig 1 (b) shows the corresponding CSMA Markov Chain. State (0,0) means that no link is transmitting, state (1,0) means that only link 1 is transmitting, and (0,1) means that only link 2 is transmitting. The state (1,1) is not feasible.

Let $r_k = \log(R_k)$. We call r_k the "transmission aggressiveness" (TA) of link k. For a given positive vector $\mathbf{r} = \{r_k, k = 1, \ldots, K\}$, the CSMA Markov chain is irreducible. Designate the stationary distribution of its feasible states x^i by $p(x^i; \mathbf{r})$. We have the following result.

Lemma 1: ([21], [22], [23]) The stationary distribution of the CSMA Markov chain has the following product-form:

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

where

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

Note that the summation $\sum_j$ is over all feasible states x^j.

Remark: The lemma holds as long as the ratio between the mean transmission time and mean backoff time of link k is $R_k = \exp(r_k), \forall k$ [21], [22], [23].

Proof: We verify that the distribution (1)-(2) satisfies the detailed balance equations (see [20]). Consider states x^i and $x^i + \mathbf{e}_k$ where $x^i_k = 0$ and $x^i_m = 0, \forall m \in \mathcal{N}(k)$. From (1), we have

$$\frac{p(x^i + \mathbf{e}_k; \mathbf{r})}{p(x^i; \mathbf{r})} = \exp(r_k) = R_k$$

which is exactly the detailed balance equation between state x^i and $x^i + \mathbf{e}_k$. Such relations hold for any two states that differ in only one element, which are the only pairs that correspond to nonzero transition rates. It follows that the distribution is invariant. ■

Note that the CSMA Markov chain is time-reversible since the detailed balance equations hold. In fact, the Markov chain is a reversible "spatial process" and its stationary distribution (1) is a Markov Random Field ([20], page 189; [24]). (This means that the state of every link k is conditionally independent of all other links, given the transmission states of its conflicting links.)

Later, we also write $p(x^i; \mathbf{r})$ as $p_i(\mathbf{r})$ for simplicity. These notations are interchangeable throughout the paper. And let $\mathbf{p}(\mathbf{r}) \in \mathcal{R}^N_+$ be the vector of all $p_i(\mathbf{r})$'s. In Fig 1, for example, the probabilities of state (0,0), (1,0) and (0,1) are $1/(1+R_1+R_2)$, $R_1/(1+R_1+R_2)$ and $R_2/(1+R_1+R_2)$ in the stationary distribution.

It follows from Lemma 1 that $s_k(\mathbf{r})$, the probability that link k transmits, is given by

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

Without loss of generality, assume that each link k has a capacity of 1. That is, if link k transmits data all the time (without contention from other links), then its service rate is 1 (unit of data per unit time). Then, $s_k(\mathbf{r})$ is also the *normalized* throughput (or service rate) with respect to the link capacity.

Even if the distributions of the waiting time and transmission time are not exponential distributed but have the same means ($1/R_k$ and 1), reference [23] shows that the stationary distribution (1) still holds. That is, the stationary distribution is insensitive.

C. Adaptive CSMA for maximal throughput

Assume i.i.d. traffic arrival at each link k with arrival rate λ_k. $\lambda_k \le 1$ is also *normalized* with respect to the link capacity 1, and thus can be viewed as the fraction of time when link k needs to be active to serve the arrival traffic. And denote the vector of arrival rates as $\lambda \in R^K_+$. Further assume that $\lambda_k > 0, \forall k$ without loss of generality, since the link(s) with zero arrival rate can be removed from the problem. We say that λ is *feasible* if and only if $\lambda = \sum_i \bar{p}_i \cdot x^i$ for some probability distribution $\bar{\mathbf{p}} \in \mathcal{R}^N_+$ satisfying $\bar{p}_i \ge 0$ and $\sum_i \bar{p}_i = 1$. That is, λ is a convex combination of the IS's, such that it is possible to serve the arriving traffic with some transmission schedule. Denote the set of feasible λ by $\bar{\mathcal{C}}$. 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 strictly feasible λ by $\mathcal{C}$. Appendix A shows that $\mathcal{C}$ is exactly the interior of $\bar{\mathcal{C}}$.

Define the following function (the "log-likelihood function" if we estimate the parameter $\mathbf{r}$ from the observation $\bar{p}_i$)

$$\begin{aligned} F(\mathbf{r}; \lambda) &:= \sum_i \bar{p}_i \log(p_i(\mathbf{r})) \\ &= \sum_i \bar{p}_i [\sum_{k=1}^K x^i_k r_k - \log(C(\mathbf{r}))] \\ &= \sum_k \lambda_k r_k - \log(\sum_j \exp(\sum_{k=1}^K x^j_k r_k)) \end{aligned}$$

where $\lambda_k = \sum_i \bar{p}_i x^i_k$ is the traffic arrival rate at link k.

Consider the following optimization problem

$$\sup_{\mathbf{r}\geq 0} F(\mathbf{r};\lambda) \ . \quad (4)$$

Since $\log(p(x^i;\mathbf{r})) \leq 0$, we have $F(\mathbf{r};\lambda) \leq 0$. Therefore $\sup_{\mathbf{r}\geq 0} F(\mathbf{r};\lambda)$ exists. Also, $F(\mathbf{r};\lambda)$ is concave in $\mathbf{r}$ [28]. We show that the following proposition holds.

Proposition 1: If $\sup_{\mathbf{r}\geq 0} F(\mathbf{r};\lambda)$ is attainable (i.e., there exists *finite* $\mathbf{r}^* \geq 0$ such that $F(\mathbf{r}^*;\lambda) = \sup_{\mathbf{r}\geq 0} F(\mathbf{r};\lambda)$), then $s_k(\mathbf{r}^*) \geq \lambda_k, \forall k$. That is, the service rate is not less than the arrival rate when $\mathbf{r} = \mathbf{r}^*$.

Proof: Let $\mathbf{d} \geq 0$ be a vector of dual variables associated with the constraints $\mathbf{r} \geq 0$ in problem (4), then the Lagrangian is $\mathcal{L}(\mathbf{r};\mathbf{d}) = F(\mathbf{r};\lambda) + \mathbf{d}^T\mathbf{r}$. At the optimal solution $\mathbf{r}^*$, we have

$$\begin{aligned}\frac{\partial \mathcal{L}(\mathbf{r}^*;\mathbf{d}^*)}{\partial r_k} &= \lambda_k - \frac{\sum_j x_k^j \exp(\sum_{k=1}^K x_k^j r_k^*)}{C(\mathbf{r}^*)} + d_k^* \\ &= \lambda_k - s_k(\mathbf{r}^*) + d_k^* = 0 \quad (5)\end{aligned}$$

where $s_k(\mathbf{r})$, according to (3), is the service rate (at stationary distribution) given $\mathbf{r}$. Since $d_k^* \geq 0$, $\lambda_k \leq s_k(\mathbf{r}^*)$. ■

Equivalently, problem (4) is the same as minimizing the Kullback–Leibler divergence (KL divergence) between the two distributions $\bar{\mathbf{p}}$ and $\mathbf{p}(\mathbf{r})$:

$$\inf_{\mathbf{r}\geq\mathbf{0}} D_{KL}(\bar{\mathbf{p}}||\mathbf{p}(\mathbf{r})) \quad (6)$$

where the KL divergence

$$\begin{aligned}D_{KL}(\bar{\mathbf{p}}||\mathbf{p}(\mathbf{r})): &= \textstyle\sum_i[\bar{p}_i \log(\bar{p}_i/p_i(\mathbf{r}))] \\ &= \textstyle\sum_i[\bar{p}_i \log(\bar{p}_i)] - F(\mathbf{r};\lambda).\end{aligned}$$

That is, we choose $\mathbf{r} \geq \mathbf{0}$ such that $\mathbf{p}(\mathbf{r})$ is the "closest" to $\bar{\mathbf{p}}$ in terms of the KL divergence.

The following condition, proved in Appendix B, ensures that $\sup_{\mathbf{r}\geq 0} F(\mathbf{r};\lambda)$ is attainable.

Proposition 2: If the arrival rate λ is strictly feasible, then $\sup_{\mathbf{r}\geq 0} F(\mathbf{r};\lambda)$ is attainable.

Combining Propositions 1 and 2, we know that for any strictly feasible λ there exists a finite $\mathbf{r}^*$ such that $s_k(\mathbf{r}^*) \geq \lambda_k, \forall k$. To see why strict feasibility is necessary, consider the network in Fig. 1. If $\lambda_1 = \lambda_2 = 0.5$ (not strictly feasible), then the service rates $s_1(\mathbf{r}) = s_2(\mathbf{r}) \to 0.5$ only when $r_1 = r_2 \to \infty$, but they cannot reach 0.5 for finite values of $\mathbf{r}$.

Since $\partial F(\mathbf{r};\lambda)/\partial r_k = \lambda_k - s_k(\mathbf{r})$, a simple gradient algorithm to solve (4) is

$$r_k(j+1) = [r_k(j) + \alpha(j)\cdot(\lambda_k - s_k(\mathbf{r}(j)))]_+, \forall k \quad (7)$$

where $j = 0,1,2,\ldots$, and $\alpha(j)$ is some (small) step size. The algorithm is easy for *distributed* implementation in wireless networks, because link k can adjust r_k based on its *local information*: arrival rate λ_k and service rate $s_k(\mathbf{r}(j))$. (If the arrival rate is larger than the service rate, then r_k should be increased, and vice versa.) Note that however, the arrival and service rates are generally random variables in actual networks, unlike in (7).

Let link k adjust r_k at time t_j, $j = 1,2,\ldots$. 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 $\mathbf{r}(j)$ as the value of $\mathbf{r}$ set at time t_j. Let $\lambda'_k(j)$ be the average arrival rate between time t_j and t_{j+1}, and let $s'_k(j)$ be the average service rate between t_j and t_{j+1}. That is, $s'_k(j) := \int_{t_j}^{t_{j+1}} x_k(\tau)d\tau/T(j+1)$, where $x_k(\tau) \in \{0,1\}$ is the state of link k at time instance τ. Note that $\lambda'_k(j)$ and $s'_k(j)$ are generally random variables. We design the following distributed algorithm.

Algorithm 1: Adjusting the TA (transmission aggressiveness) in CSMA

At time t_{j+1} where $j = 0,1,2,\ldots$, let

$$r_k(j+1) = [r_k(j) + \alpha(j)\cdot(\lambda'_k(j) - s'_k(j))]_D, \forall k \quad (8)$$

where $\alpha(j) > 0$ is the step size, and $[\cdot]_D$ means the projection to the set $D := [0, r_{max}]$ where $r_{max} > 0$. We allow $r_{max} = +\infty$, in which case the projection is the same as $[\cdot]_+$.[6] In the next section and Appendix C, we will discuss the convergence and stability property of Algorithm 1 under different settings of $\alpha(j), T(j)$ and r_{max}.

D. Convergence and stability

Reference [35] provides some stability results of the following algorithm extended from Algorithm 1. The intuition is that one can make $\mathbf{r}$ change slowly (i.e., "quasi-static") to allow the CSMA Markov chain to approach its stationary distribution (and thus obtaining good estimation of $s_k(\mathbf{r})$). This allows the separation of time scales of the dynamics of $\mathbf{r}(j)$ and the CSMA Markov chain. The extended algorithm is

$$r_k(j+1) = [r_k(j) + \alpha(j)\cdot(\lambda'_k(j) + h(r_k(j)) - s'_k(j))]_D \quad (9)$$

where $D := [0, r_{max}]$ and the function $h(\cdot) \geq 0$. If $h(\cdot) = 0$, then algorithm (9) reduces to Algorithm 1. If $h(\cdot) > 0$, then algorithm (9) "pretends" to serve some arrival rates higher than the actual ones. In Appendix C, we state some results in [35] (which includes the detailed proofs). In summary, (i) with properly-chosen decreasing step sizes and increasing adjustment periods (e.g., $\alpha(j) = 1/[(j+2)\log(j+2)]$, $T(j) = j+2$) and function $h(\cdot)$, and with $r_{max} = +\infty$, the vector $\mathbf{r}(j)$ converges and the algorithm is throughput-optimal; (ii) with properly-chosen constant step sizes $\alpha(j) = \alpha, \forall j$, adjustment periods $T(j) = T, \forall j$, one can arbitrarily approximate the maximal throughput.

In a related work [17], Liu et al. carried out a convergence analysis, using a differential-equation method, of a utility maximization algorithm extended from [2] (see also section IV for the algorithm).

E. Discussion

(1) Since optimal scheduling is NP complete with the general interference model in this paper, the complexity is reflected in the convergence time of CSMA Markov chain. In [35], the worst case upper-bound used to quantify the time for the CSMA Markov chain to approach its stationary distribution (i.e., the mixing time) is exponential in K. Typical wireless

[6] A subtle point: If in period $j+1$ (for any j), the queue of link k' becomes empty, then link k' still transmits dummy packets with TA $r_{k'}(j)$ until t_{j+1}. This ensures that the (ideal) average service rate is still $s_k(\mathbf{r}(j))$ for all k.

networks, however, may not be the worst case. For example, in a network where all links conflict, the CSMA Markov chain can be shown to mix fast.

(2) There is some resemblance between the above algorithm (in particular the CSMA Markov chain) and simulated annealing (SA) [16]. SA is an optimization technique that utilizes time-reversible Markov chains to find a maximum of a function. SA can be used, for example, to find the Maximal-Weighted IS (MWIS) which is needed in Maximal-Weight Scheduling. However, note that our algorithm does not try to find the MWIS via SA. Instead, the stationary distribution of the CSMA Markov chain with a properly-chosen $\mathbf{r}^*$ is sufficient to support any vector of strictly feasible arrival rates (Proposition 1).

III. The Primal-dual relationship

In the previous section we have described the adaptive CSMA algorithm to support any strictly-feasible arrival rates. For joint scheduling and flow control, however, directly using the above expression of service rate (3) will lead to a non-convex problem. This section gives another look at the problem and also helps to avoid the difficulty.

Rewrite (4) as

$$\begin{array}{ll} \max_{\mathbf{r},\mathbf{z}} & \{\sum_k \lambda_k r_k - \log(\sum_j \exp(h_j))\} \\ \text{s.t.} & h_j = \sum_{k=1}^K x_k^j r_k, \forall j \\ & r_k \geq 0, \forall k. \end{array} \quad (10)$$

For each $j = 1, 2, \ldots, N$, associate a dual variable u_j to the constraint $h_j = \sum_{k=1}^K x_k^j r_k$. Write the vector of dual variables as $\mathbf{u} \in \mathcal{R}_+^N$. Then it is not difficult to find the dual problem of (10) as follows. (We omit the computation here due to the limit of space.)

$$\begin{array}{ll} \max_{\mathbf{u}} & -\sum_i u_i \log(u_i) \\ \text{s.t.} & \sum_i (u_i \cdot x_k^i) \geq \lambda_k, \forall k \\ & u_i \geq 0, \sum_i u_i = 1. \end{array} \quad (11)$$

where the objective function is the entropy of the distribution $\mathbf{u}$, $H(\mathbf{u}) := -\sum_i u_i \log(u_i)$. [7]

Also, if for each k, we associate a dual variable r_k to the constraint $\sum_i (u_i \cdot x_k^i) \geq \lambda_k$ in problem (11), then one can compute that the dual problem of (11) is the original problem $\max_{\mathbf{r} \geq \mathbf{0}} F(\mathbf{r}; \lambda)$ (This is shown in Appendix B as a by-product of the proof of Proposition 2). This is not surprising, since in convex optimization, the dual problem of dual problem is often the original problem.

What is interesting is that both $\mathbf{r}$ and $\mathbf{u}$ have concrete physical meanings. We have seen that r_k is the TA of link k. Also, u_i can be regarded as the stationary probability of state i in the CSMA Markov chain given the dual variable $\mathbf{r}$. This observation will be useful in later sections. A convenient way to guess this is by observing the constraint $\sum_i (u_i \cdot x_k^i) \geq \lambda_k$. If u_i is the probability of state i, then the constraint simply means that the service rate of link k, $\sum_i (u_i \cdot x_k^i)$, is larger than the arrival rate.

[7] In fact, there is a more general relationship between ML estimation problem such as (4) and Maximal-Entropy problem such as (11) [29].

Proposition 3: Given some (finite) TA's of the links (that is, given the dual variable $\mathbf{r}$ of problem (11)), the stationary distribution of the CSMA Markov chain maximizes the partial Lagrangian $\mathcal{L}(\mathbf{u}; \mathbf{r}) = -\sum_i u_i \log(u_i) + \sum_k r_k (\sum_i u_i \cdot x_k^i - \lambda_k)$ over all possible distributions $\mathbf{u}$. Also, Algorithm (7) can be viewed as a subgradient algorithm to update the dual variable $\mathbf{r}$ in order to solve problem (11).

Proof: Given some finite dual variables $\mathbf{r}$, a partial Lagrangian of problem (11) is

$$\mathcal{L}(\mathbf{u}; \mathbf{r}) = -\sum_i u_i \log(u_i) + \sum_k r_k (\sum_i u_i \cdot x_k^i - \lambda_k). \quad (12)$$

Denote $\mathbf{u}^*(\mathbf{r}) = \arg\max_{\mathbf{u}} \mathcal{L}(\mathbf{u}; \mathbf{r})$, where $\mathbf{u}$ is a distribution. Since $\sum_i u_i = 1$, if we can find some w, and $\mathbf{u}^*(\mathbf{r}) > 0$ (i.e., in the interior of the feasible region) such that

$$\frac{\partial \mathcal{L}(\mathbf{u}^*(\mathbf{r}); \mathbf{r})}{\partial u_i} = -\log(u_i^*(\mathbf{r})) - 1 + \sum_k r_k x_k^i = w, \forall i,$$

then $\mathbf{u}^*(\mathbf{r})$ is the desired distribution. The above conditions are

$$u_i^*(\mathbf{r}) = \exp(\sum_k r_k x_k^i - w - 1), \forall i. \text{ and } \sum_i u_i^*(\mathbf{r}) = 1.$$

By solving the two equations, we find that $w = \log[\sum_j \exp(\sum_k r_k x_k^j)] - 1$ and

$$u_i^*(\mathbf{r}) = \frac{\exp(\sum_k r_k x_k^i)}{\sum_j \exp(\sum_k r_k x_k^j)}, \forall i \quad (13)$$

satisfy the conditions.

Note that in (13), $u_i^*(\mathbf{r})$ is exactly the stationary probability of state i in the CSMA Markov chain given the TA $\mathbf{r}$ of all links. That is, $u_i^*(\mathbf{r}) = p(x^i; \mathbf{r}), \forall i$ (cf. (1)). So Algorithm (7) is a subgradient algorithm to search for the optimal dual variable. Indeed, given $\mathbf{r}$, $u_i^*(\mathbf{r})$ maximizes $\mathcal{L}(\mathbf{u}; \mathbf{r})$; then, $\mathbf{r}$ can be updated by the subgradient algorithm (7), which is the deterministic version of Algorithm 1. The whole system is trying to solve problem (11) or (4). ■

Let $\mathbf{r}^*$ be the optimal vector of dual variables of problem (11). From the above computation, we see that $\mathbf{u}^*(\mathbf{r}^*) = \mathbf{p}(\mathbf{r}^*)$, the optimal solution of (11), is a product-form distribution. Also, $\mathbf{p}(\mathbf{r}^*)$ can support the arrival rates λ because it is feasible to (11). This is another way to look at Proposition 1.

IV. Joint scheduling and rate control

Now, we combine end-to-end rate control with the CSMA scheduling algorithm to achieve fairness among competing flows as well as maximal throughput. Here, the input rates are distributedly adjusted by the source of each flow.

A. Formulation

Assume there are M flows, and let m be their index ($m = 1, 2, \ldots, M$). Define $a_{mk} = 1$ if flow m uses link k, and $a_{mk} = 0$ otherwise. Let f_m be the rate of flow m, and $v_m(f_m)$ be the "utility function" of this flow, which is assumed to be increasing and strictly concave. Assume all links have the same PHY data rates (it is easy to extend the algorithm to different PHY rates).

Assume that each link k maintains a separate queue for each flow that traverses it. Then, the service rate of flow m by link k, denoted by s_{km}, should be no less than the incoming rate of flow m to link k. For flow m, if link k is its first link (i.e., the source link), we say $\delta(m) = k$. In this case, the constraint is $s_{km} \geq f_m$. If $k \neq \delta(m)$, denote flow m's upstream link of link k by $up(k,m)$, then the constraint is $s_{km} \geq s_{up(k,m),m}$, where $s_{up(k,m),m}$ is equal to the incoming rate of flow m to link k. We also have $\sum_i u_i \cdot x_k^i \geq \sum_{m:a_{mk}=1} s_{km}, \forall k$, i.e., the total service rate of link k is not less than the sum of all flow rates on the link.

Then, consider the following optimization problem:

$$\begin{array}{ll} \max_{\mathbf{u},\mathbf{s},\mathbf{f}} & -\sum_i u_i \log(u_i) + \beta \sum_{m=1}^M v_m(f_m) \\ \text{s.t.} & s_{km} \geq 0, \forall k, m : a_{mk} = 1 \\ & s_{km} \geq s_{up(k,m),m}, \forall m, k : a_{mk} = 1, k \neq \delta(m) \\ & s_{km} \geq f_m, \forall m, k : k = \delta(m) \\ & \sum_i u_i \cdot x_k^i \geq \sum_{m:a_{mk}=1} s_{km}, \forall k \\ & u_i \geq 0, \sum_i u_i = 1. \end{array} \tag{14}$$

where $\beta > 0$ is a weighting factor.

Notice that the objective function is not exactly the total utility, but it has an extra term $-\sum_i u_i \log(u_i)$. As will be further explained in section IV-B, when β is large, the "importance" of the total utility dominates the objective function of (14). (This is similar in spirit to the weighting factor used in [15].) As a result, the solution of (14) approximately achieves the maximal utility. Associate dual variables $q_{km} \geq 0$ to the 2nd and 3rd lines of constraints of (14). Then a partial Lagrangian (subject to $s_{km} \geq 0$, $\sum_i u_i \cdot x_k^i \geq \sum_{m:a_{mk}=1} s_{km}$ and $u_i \geq 0, \sum_i u_i = 1$) is

$$\begin{array}{rl} & \mathcal{L}(\mathbf{u},\mathbf{s},\mathbf{f};\mathbf{q}) \\ = & -\sum_i u_i \log(u_i) + \beta \sum_{m=1}^M v_m(f_m) \\ & + \sum_{m,k:a_{mk}=1,k\neq\delta(m)} q_{km}(s_{km} - s_{up(k,m),m}) \\ & + \sum_{m,k:,k=\delta(m)} q_{km}(s_{km} - f_m) \\ = & -\sum_i u_i \log(u_i) \\ & +\beta \sum_{m=1}^M v_m(f_m) - \sum_{m,k:k=\delta(m)} q_{km} f_m \\ & + \sum_{k,m:a_{mk}=1} s_{km}[(q_{km} - q_{down(k,m),m})] \end{array} \tag{15}$$

where $down(k,m)$ means flow m's downstream link of link k (Note that $down(up(k,m),m) = k$). If k is the last link of flow m, then define $q_{down(k,m),m} = 0$.

Fix the vectors $\mathbf{u}$ and $\mathbf{q}$ first, we solve for s_{km} in the sub-problem

$$\begin{array}{ll} \max_{\mathbf{s}} & \sum_{k,m:a_{mk}=1} s_{km}[(q_{km} - q_{down(k,m),m})] \\ \text{s.t.} & s_{km} \geq 0, \forall k, m : a_{mk} = 1 \\ & \sum_{m:a_{mk}=1} s_{km} \leq \sum_i (u_i \cdot x_k^i), \forall k. \end{array} \tag{16}$$

The solution is easy to find (similar to [1] and related references therein): at link k, denote $z_k := \max_{m:a_{mk}=1}(q_{km} - q_{down(k,m),m})$. (i) If $z_k > 0$, then for a $m' \in \arg\max_{m:a_{mk}=1}(q_{km} - q_{down(k,m),m})$, let $s_{km'} = \sum_i (u_i \cdot x_k^i)$ and let $s_{km} = 0, \forall m \neq m'$. In other words, link k serves the flow with the maximal back-pressure $q_{km} - q_{down(k,m),m}$. (ii) If $z_k \leq 0$, then let $s_{km}(j) = 0, \forall m$, i.e., link k does not serve any flow (and transmit dummy packets instead when it has the opportunity to transmit). Since the value of $q_{down(k,m),m}$ can be obtained from a one-hop neighbor, this algorithm is distributed.

Plug the solution of (16) back into (15), we get

$$\begin{array}{rcl} \mathcal{L}(\mathbf{u},\mathbf{f};\mathbf{q}) & = & [-\sum_{i=1} u_i \log(u_i) + \sum_k (z_k)_+ (\sum_i u_i \cdot x_k^i)] \\ & & +[\beta \sum_{m=1}^M v_m(f_m) - \sum_{m,k:k=\delta(m)} q_{km} f_m] \end{array}$$

where $z_k := \max_{m:a_{mk}=1}(q_{km} - q_{down(k,m),m})$ is the maximal back-pressure at link k. So a distributed algorithm to solve (14) is as follows. For simplicity, assume that $v_m'(0) \leq V < \infty, \forall m$, i.e., the derivative of all utility functions at 0 is bounded by some $V < \infty$.

Algorithm 2: Joint scheduling and rate control

Initially, assume that all queues are empty, and let $q_{km}(0) = 0, \forall k, m$. Here we use $\alpha(j) = \alpha, T(j) = T, \forall j$. The variables $\mathbf{q}, \mathbf{f}, \mathbf{r}$ are iteratively updated at time t_j, $j = 0, 1, 2, \ldots$: Let $\mathbf{q}(j), \mathbf{f}(j), \mathbf{r}(j)$ be the values set at time t_j. Denote by $s_{km}^{'}(j)$ the empirical average service rate of flow m at link k in period $j+1$ (i.e., the time between t_j and t_{j+1}).

- Scheduling: In period $j+1$, link k lets its TA be $r_k(j) = [z_k(j)]_+$ in the CSMA operation, where $z_k(j) = \max_{m:a_{mk}=1}(q_{km}(j) - q_{down(k,m),m}(j))$. (This is because, given $\mathbf{z}(j)$, the optimal $\mathbf{u}$ (that maximizes $\mathcal{L}(\mathbf{u},\mathbf{f};\mathbf{q}(j))$ over $\mathbf{u}$) is the stationary distribution of the CSMA Markov Chain with $r_k(j) = [z_k(j)]_+$, similar to the proof of Proposition 3.) When link k gets the opportunity to transmit, (i) if $z_k(j) > 0$, it serves a flow $m' \in \arg\max_{m:a_{mk}=1}(q_{km}(j) - q_{down(k,m),m}(j))$; (ii) if $z_k(j) \leq 0$, then it transmits dummy packets (which are not counted when computing $s_{km}^{'}(j)$).
- Rate control: For each flow m, if link k is its source link, the transmitter of link k lets the flow rate in period $j+1$ be $f_m(j) = \arg\max_{f_m' \in [0,1]} \{\beta \cdot v_m(f_m') - q_{km}(j) \cdot f_m'\}$. (This maximizes $\mathcal{L}(\mathbf{u},\mathbf{f};\mathbf{q}(j))$ over $\mathbf{f}$.)
- The dual variables q_{km} (maintained by the transmitter of each link) are updated by a sub-gradient algorithm. At time t_{j+1}, let $q_{km}(j+1) = [q_{km}(j) + \alpha(s_{up(k,m),m}^{'}(j) - s_{km}^{'}(j))]_+$ if $k \neq \delta(m)$; and $q_{km}(j+1) = [q_{km}(j) + \alpha(f_m(j) - s_{km}^{'}(j))]_+$ if $k = \delta(m)$. (By doing this, $q_{km} \propto Q_{km}$ roughly, where Q_{km} is the queue length of flow m at link k.)

Remark 1: As $T \to \infty$ and $\alpha \to 0$, Algorithm 2 approximates the "ideal" algorithm that solves (14), due to the convergence of the CSMA Markov chain in each period. A bound of the achievable utility of Algorithm 2, compared to the optimal total utility $\bar{W}$ defined in (17) is given in Appendix E. The bound, however, is not very tight, since our simulation shows good performance without a very large T or a very small α.

Remark 2: In Appendix F, we show that by using similar techniques, the adaptive CSMA algorithm can be combined with optimal routing, anycast or multicast. So it is a modular MAC-layer protocol which can work with other protocols in the transport layer and the network layer.

B. Approaching the maximal utility

Notice that $-\sum_i u_i \log(u_i)$, the entropy of the distribution $\mathbf{u}$, is bounded. Indeed, since there are $N \leq 2^K$ possible states, one has $0 \leq -\sum_i u_i \log(u_i) \leq \log N \leq \log 2^K = K \cdot \log 2$.

Therefore, as mentioned earlier, when β is large, the "importance" of the total utility dominates the objective function of (14). So the solution of (14) approximately achieves the maximal utility. Denote the highest total utility achievable as $\bar{W}$, i.e.,

$$\bar{W} := \max_{\mathbf{u},\mathbf{s},\mathbf{f}} \sum_m v_m(f_m) \tag{17}$$

subject to the same constraints as in (14). Assume that $\mathbf{u} = \bar{\mathbf{u}}$ when (17) is solved. Also, assume that in the optimal solution of (14), $\mathbf{f} = \hat{\mathbf{f}}$ and $\mathbf{u} = \hat{\mathbf{u}}$. We prove the following bound in Appendix D.

Proposition 4: The difference between the total utility ($\sum_{m=1}^{M} v_m(\hat{f}_m)$) resulting from solving (14) and the maximal total utility $\bar{W}$ is bounded. The bound of difference decreases with the increase of β. In particular,

$$\bar{W} - (K \cdot \log 2)/\beta \le \sum_m v_m(\hat{f}_m) \le \bar{W}. \tag{18}$$

V. Reducing the queueing delay

Consider a strictly feasible arrival rate vector λ in the scheduling problem in section II. With Algorithm 1, the long-term average service rates are in general not strictly higher than the arrival rates, so traffic suffers from queueing delay when traversing the links. To reduce the delay, consider a modified version of problem (11):

$$\begin{array}{ll} \max_{\mathbf{u},\mathbf{w}} & -\sum_i u_i \log(u_i) + c\sum_k \log(w_k) \\ \text{s.t.} & \sum_i (u_i \cdot x_k^i) \ge \lambda_k + w_k, \forall k \\ & u_i \ge 0, \sum_i u_i = 1 \\ & 0 \le w_k \le \bar{w}, \forall k \end{array} \tag{19}$$

where $0 < c < 1$ is a small constant. Note that we have added the new variables $w_k \in [0, \bar{w}]$ (where $\bar{w}$ is a constant upper bound), and require $\sum_i u_i \cdot x_k^i \ge \lambda_k + w_k$. In the objective function, the term $c \cdot \log(w_k)$ is a penalty function to avoid w_k being too close to 0.

Since λ is in the interior of the capacity region, there is a vector λ' also in the interior and satisfying $\lambda' > \lambda$ component-wise. So there exist $\mathbf{w}' > 0$ and $\mathbf{u}'$ (such that $\sum_i u_i' x_k^i = \lambda_k' := \lambda_k + w_k', \forall k$) satisfying the constraints. Therefore, in the optimal solution, we have $w_k^* > 0, \forall k$ (otherwise the objective function is $-\infty$, smaller than the objective value that can be achieved by $\mathbf{u}'$ and $\mathbf{w}'$). Thus $\sum_i u_i^* \cdot x_k^i \ge \lambda_k + w_k^* > \lambda_k$. This means that the service rate is strictly larger than the arrival rate, bringing the extra benefit that the queue lengths tend to decrease to 0.

Similar to section III, we form a partial Lagrangian (with $\mathbf{y} \ge 0$ as dual variables)

$$\begin{aligned} \mathcal{L}(\mathbf{u}, \mathbf{w}; \mathbf{y}) &= -\sum_i u_i \log(u_i) + c\sum_k \log(w_k) + \\ &\quad \sum_k [y_k(\sum_i u_i \cdot x_k^i - \lambda_k - w_k)] \\ &= [-\sum_i u_i \log(u_i) + \sum_k (y_k \sum_i u_i \cdot x_k^i)] + \\ &\quad \sum_k [c \cdot \log(w_k) - y_k w_k] - \sum_k (y_k \lambda_k). \end{aligned} \tag{20}$$

Note that the only difference from (12) is the extra term $\sum_k [c \cdot \log(w_k) - y_k w_k]$. Given $\mathbf{y}$, the optimal $\mathbf{w}$ is $w_k = \min\{c/y_k, \bar{w}\}, \forall k$, and the optimal $\mathbf{u}$ is the stationary distribution of the CSMA Markov Chain with $\mathbf{r} = \mathbf{y}$. Therefore the (sub)gradient algorithm to update $\mathbf{y}$ is $y_k \leftarrow y_k + \alpha(\lambda_k + w_k - s_k(\mathbf{y}))$.

Since $\mathbf{r} = \mathbf{y}$, we have the following localized algorithm at link k to update r_k. Notice its similarity to Algorithm 1.

Algorithm 3: Enhanced Algorithm 1 to reduce queueing delays

At time t_{j+1} where $j = 0, 1, 2, \ldots$, let

$$r_k(j+1) = [r_k(j) + \alpha(j) \cdot (\lambda_k'(j) + \min\{c/r_k(j), \bar{w}\} - s_k'(j))]_D \tag{21}$$

for all k, where $\alpha(j)$ is the step size, and $D = [0, r_{max}]$ where r_{max} can be $+\infty$. As before, even when link k' has no backlog (i.e., zero queue length), we let it send dummy packet with its current aggressiveness $r_{k'}$. This ensures that the average service rate of link k is $s_k(\mathbf{r}(j))$ for all k.

Since Algorithm 3 "pretends" to serve some arrival rates higher than the actual arrival rates (due to the positive term $\min\{c/r_k(j), \bar{w}\}$, Q_k is not only stable, but also tends to be small. The convergence and stability properties of Algorithm 3 when $r_{max} = \infty$ are discussed in (i) of Appendix C. If $r_{max} < \infty$, the properties are similar to those in (ii) of Appendix C.

For the end-to-end utility maximization (without a given arrival rate vector), a simple way to reduce the delay, similar to [37], is as follows. In item 2 ("rate control") of Algorithm 2, let the actual flow rate be $\rho \cdot f_m(j)$ where ρ is slightly smaller than 1, and keep other parts of the algorithm unchanged. Then, each link provides a service rate higher than the actual arrival rate. So the delay is reduced with a small cost in the flow rates.

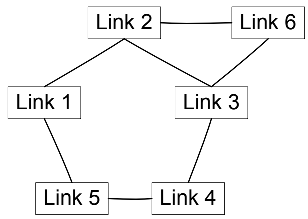
VI. Simulations

A. CSMA scheduling: i.i.d. input traffic with fixed average rates

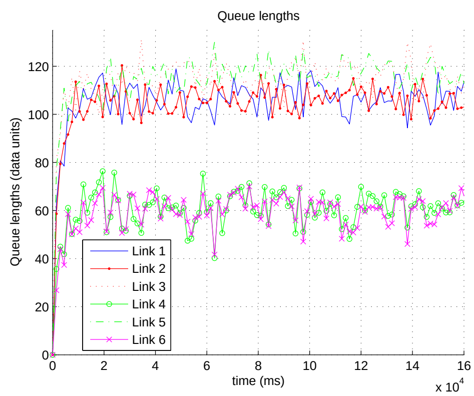
In our C++ simulations, the transmission time of all links is exponentially distributed with mean 1ms, and the backoff time of link k is exponentially distributed with mean $1/\exp(r_k)$ ms. Assume that the capacity of each link is 1(data unit)/ms. Initially, all queues are empty, and the initial value of r_k is 0 for all k. r_k is then adjusted using Algorithm 1 once every $T = 5ms$ (i.e., $T(j) = T, \forall j$), with a constant step size $\alpha(j) = \alpha = 0.23, \forall j$.

There are 6 links in "Network 1", whose LCG is shown in Fig. 2 (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. Fig. 2 (b) shows the evolution of the queue lengths using Algorithm 1 with $r_{max} = 8$. They are stable despite some oscillations. The vector $\mathbf{r}$ is not shown since in this simulation, it is α/T times the queue lengths. Fig. 2 (c) shows the evolution of queue lengths using Algorithm 3 with $c = 0.01$, $\bar{w} = 0.02$ and $r_{max} = 8$, which drives the queue lengths to around zero, thus significantly reducing the queueing delays.

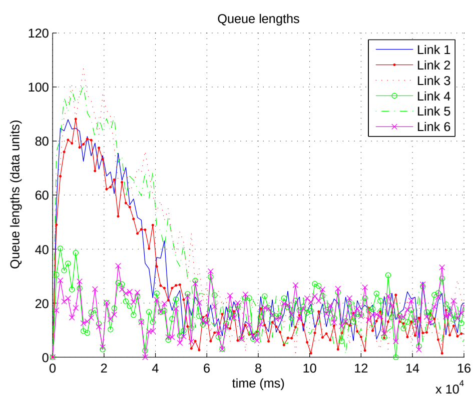
Fig 3 shows the results of Algorithm 3 with $\alpha(j) = 0.46/[(2+j/1000)\log(2+j/1000)]$ and $T(j) = (2+j/1000)$



(a) Link Contention Graph

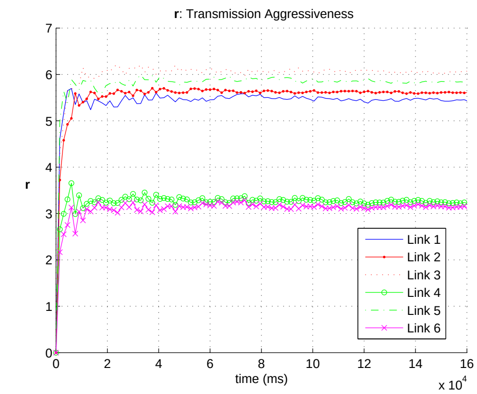


(b) Queue lengths, with constant step size. The vector **r** is not shown since it is proportional to the queue lengths.

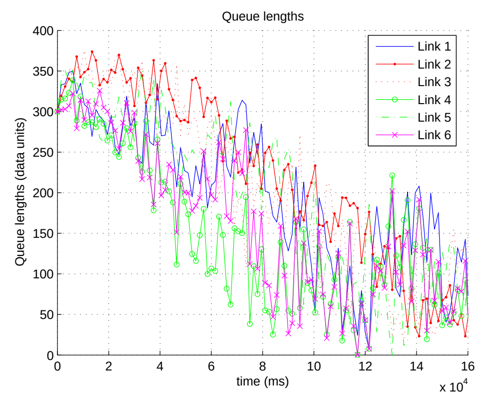


(c) Queue lengths (with Algorithm 3)

Fig. 2: Adaptive CSMA Scheduling with fixed input rates (Network 1)



(a) The vector **r**



(b) Queue lengths

Fig. 3: Decreasing step sizes

ms, which satisfy the conditions for convergence in [35]. The constants $c = 0.01$, $\bar{w} = 0.02$, and $r_{max} = \infty$. To show the negative drift of queues, assume that initially, all queue lengths are 300 data units in Fig 3. We see that the TA vector $\mathbf{r}$ converges (Fig 3 (a)), and the queues tend to decrease and are stable (Fig 3 (b)). However, there are more oscillations in the queue lengths than the case with constant step size. This is because when $\alpha(j)$ becomes smaller when j is large, $\mathbf{r}(j)$ becomes less responsive to the variations of queue lengths.

B. Joint scheduling and rate control

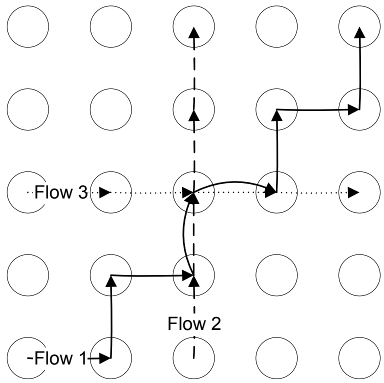
In Fig 4, we simulate a more complex network ("Network 2"). We also go one step further than Network 1 by giving the actual locations of the nodes, not only the LCG. Fig 4 (a) shows the network topology, where each circle represents a node. The nodes are arranged in a grid for convenience, and the distance between two adjacent nodes (horizontally or vertically) is 1. Assume that the transmission range is 1, so that a link can only be formed by two adjacent nodes. Assume that two links cannot transmit simultaneously if there are two nodes, one in each link, being within a distance of 1.1 (In IEEE 802.11, for example, DATA and ACK packets are transmitted in opposite directions. This model considers the interference among the two links in both directions). The paths of 3 multi-hop flows are plotted. The utility function of each flow is $v_m(f_m) = \log(f_m + 0.01)$. The weighting factor is $\beta = 3$. (Note that the input rates are adjusted by the flow control algorithm instead of being specified as in the last subsection.)

Fig 4 (b) shows the evolution of the flow rates, using Algorithm 2 with $T = 5ms$ and $\alpha = 0.23$. We see that they become relatively constant after an initial convergence. By directly solving (17) centrally, we find that the theoretical optimal flow rates for the three flows are 0.11, 0.134 and 0.134 (data unit/ms), very close to the simulation results. The queue lengths are also stable but not shown here due to the limit on space.

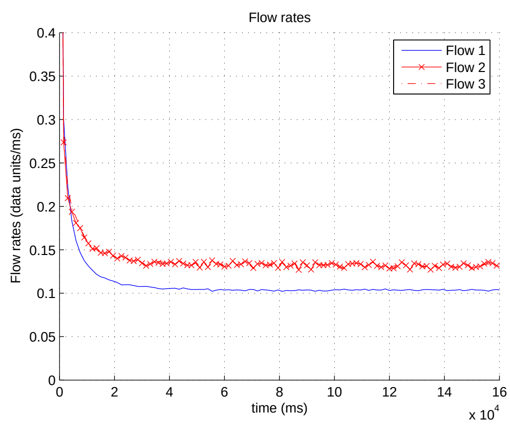
VII. Implementation Considerations in 802.11 Networks

A. Packet Collisions

In the idealized CSMA model we used, the distribution of backoff times is continuous and there is no collision. This



(a) Network 2 and flow directions



(b) Flow rates

Fig. 4: Flow rates in Network 2 (Grid Topology) with Joint scheduling and rate control

allows us to focus on the scheduling problem without worrying about the contention resolution problem. The resulting performance can serve as a benchmark. However in practice, the backoff time is usually a multiple of mini-slots, where each mini-slot cannot be arbitrarily small because the sensing time is not zero. Therefore collisions occur given the discrete distribution of backoff times. In this section we consider this practical issue and discuss alternative algorithms (for 802.11 networks) which are related to the above algorithms with idealized CSMA.

As mentioned earlier, we have recently studied a tractable model in [27] that explicitly considered collisions in wireless network without hidden nodes. Moreover, similar algorithms (with probe packets such as RTS/CTS) were proposed in [27] that can approach the maximal throughput and utility by adjusting the mean transmission times with fixed mean backoff times.

In [17] [2], etc, it is noted that by using small transmission probability in each minislot (which increases the backoff times), and correspondingly increasing the transmission times, the collision probability becomes small, in which case the actual CSMA with collisions can be approximated by the idealized CSMA.

In [30], a new protocol is proposed to deal with collisions. The protocol has control phases and data phases. Collisions only occur in the control phase, but not in the data phase. The same product-form distribution (1) can be obtained for the data phase, which is then used to achieve the maximal throughput. Meanwhile, the overhead of the control phase can be easily quantified.

In the following, we discuss how to use algorithms in this paper with collisions in mind.

1) Relationship of TA and the contention window in 802.11: Assume that for link k the average transmission time is T. Then the average backoff time is T/R_k. Denote by W_k the contention window (CW) that gives the same average backoff time. (Recall that the distribution of the backoff time is not important, as long as it has the correct mean.) Since a random number is uniformly picked from 0 to $W_k - 1$, the average backoff time is $t_m \cdot (W_k - 1)/2$, where t_m is the length of a mini-slot. (For simplicity, we do not consider the Binary Exponential Backoff, or "BEB", in this calculation.) Equating the two quantities gives

$$W_k = \frac{T}{R_k}\frac{2}{t_m} + 1. \quad (22)$$

We know that larger CW's lead to lower collision probabilities. By equation (22), for given R_k's, small mini-slot t_m or large transmission time T can lead to large CW. (If $t_m \to 0$ or $T \to +\infty$, then collisions can be ignored and we return to the previous model.) However, t_m is limited by the speed of light and the sensing time. The mean transmission time T can be made large, but should not be too large in practice since that will increase access delays. So, here we impose an upper bound, r_{max}, to all r_k's. This gives W_k's a lower bound: $W_k \geq 2T/(\exp(r_{max}) \cdot t_m) + 1$. For example, assume $T = 1ms$. Recall that a mini-slot in 802.11a is $t_m := 9\mu s$. If we require that $r_k \leq r_{max} = 2$, then $W_k \geq 31$. These values result in reasonably low collision probabilities if the number of nodes in a collision domain is not too high [33].

Although the upper bound $r_{max} = 2$ seems small, it can actually achieve a large portion of the capacity region. Consider the simple network in Fig. 1, where the throughput of the two links are $s_1(\mathbf{r}) = R_1/(1+R_1+R_2)$ and $s_2(\mathbf{r}) = R_2/(1+R_1+R_2)$ (for simplicity, here we temporarily assume that collisions are negligible due to the large CW's). The capacity region is $\mathcal{C} = \{(\lambda_1, \lambda_2)|\lambda_1 + \lambda_2 < 1\}$. If $r_1 = r_2 = r_{max}$, then the total throughput is $2 \cdot \exp(2)/[1 + \exp(2) + \exp(2)] \approx 0.937$, not far from the maximal total throughput 1.

2) A condition that ensures bounded TA: In the following, we show that by properly choosing the weighting factor β of the total utility in Algorithm 2, it can be guaranteed that every r_k is smaller than r_{max} at all time, if certain conditions are satisfied. (In [34], a similar approach is used to control the amount of backlog in the network.)

Proposition 5: Assume that the utility function $v_m(f_m)$ (strictly concave) satisfies $v'_m(0) \leq V < \infty, \forall m$. Denote by L as the largest number of hops of a flow in the network. Then by setting $\beta = [r_{max} - (2L-1) \cdot \alpha]/V$ in Algorithm 2, we have $r_k \leq r_{max}, \forall k$ at all time.

Proof: According to Algorithm 2, the source of flow m solves $f_m(j) = \arg\max_{f'_m} \{\beta \cdot v_m(f'_m) - q_{\delta(m),m}(j) \cdot f'_m\}$. It is easy to see that if $q^m_{\delta(m)}(j) \geq \beta \cdot V$, then $f_m(j) = 0$, i.e., the source stops sending data. Thus $q^m_{\delta(m)}(j+1) \le q^m_{\delta(m)}(j)$. If $q^m_{\delta(m)}(j) < \beta \cdot V$, then $q^m_{\delta(m)}(j+1) \le q^m_{\delta(m)}(j) + \alpha < \beta \cdot V + \alpha$. Since initially $q_{km}(0) = 0, \forall k, m$, by induction, we have

$$q^m_{\delta(m)}(j) \le \beta \cdot V + \alpha, \forall j, m. \quad (23)$$

Denote $b_{km}(j) := q_{km}(j) - q_{down(k,m),m}(j)$. In Algorithm 2, no matter whether flow m has the maximal back-pressure at link k, the actual average service rate $s'_{km}(j) = 0$ if $b_{km}(j) \le 0$. That is, $s'_{km}(j) > 0$ only if $b_{km}(j) > 0$. Since $s'_{km}(j) \le 1$, by item 3 of Algorithm 2, $q_{down(k,m),m}(j+1) \le q_{down(k,m),m}(j) + \alpha$ and $q_{km}(j+1) \ge q_{km}(j) - \alpha$. Then, if $b_{km}(j) > 0$, we have $b_{km}(j+1) \ge b_{km}(j) - 2\alpha > -2\alpha$. If $b_{km}(j) \le 0$, then $b_{km}(j+1) \ge b_{km}(j)$. Since $b_{km}(0) = 0$, by induction, we have

$$b_{km}(j) \ge -2\alpha, \forall j, k, m. \quad (24)$$

Since $\sum_{k:a_{mk}=1} b_{km}(j) = q^m_{\delta(m)}(j)$, combined with (23) and (24), we have $b_{km}(j) \le \beta \cdot V + \alpha + 2\alpha \cdot (L-1)$. Since $\beta = [r_{max} - (2L-1) \cdot \alpha]/V$, $b_{km}(j) \le r_{max}, \forall j, k, m$. ■

B. Discrete TA and a real-world implementation

Although r_k is continuous in our model, one may find it convenient to quantize r_k into a set of discrete values in a real implementation. Each discrete value corresponds to a different contention window (a smaller r_k corresponds to a larger CW), and this can be easily mapped to the "service classes" in IEEE 802.11e. Note that here the prioritization is based on the back-pressure instead of service type originally defined in 802.11e. Indeed, in [36], a similar protocol is implemented with 802.11e hardware and it shows superior performance compared to normal 802.11. (Different from our work, however, [36] only focuses on implementation study. Also, the CW's there are set in a more heuristic way.)

In the following simulation, we set the discrete TA (denoted by $r_{D,k}(j)$) as follows by quantizing the continuous TA, $r_k(j)$, computed by Algorithm 2:

- If $r_k(j) \ge r_{max}$, then let $r_{D,k}(j) = r_{max}$. This corresponds to the first class. Then, for $i = 2, 3, \ldots, n_c$, if $r_{max} - (i-1)G \le r_k(j) < r_{max} - (i-2)G$, then let $r_{D,k}(j) = r_{max} - (i-1)G$, where G is the granularity between adjacent classes, and n_c is the number of classes. In the simulation, we let $r_{max} = 2$, $n_c = 6$, and $G = \log(2)$ (thus, the CW of class $j+1$ is roughly twice the CW of class j).
- Define the minimal TA $r_{min} := r_{max} - (n_c - 1)G$. If $r_k(j) < r_{min}$, then do not transmit packets at all. This is a good approximation since the CW would be quite large with r_{min} (about 1000). Since transmissions are suspended, the back-pressure tends to increase. The link will resume transmission after $r_k(j)$ goes above r_{min}.

The upper figure in Fig. 5 shows that the resulting flow rates and their fluctuations are similar to those with continuous $\mathbf{r}$ (lower figure). (Collisions and BEB *are* simulated here.) This indicates that the algorithm is robust to the discretization of $\mathbf{r}$. So using a few prioritized "classes" with different CW's is enough to provide good performance.

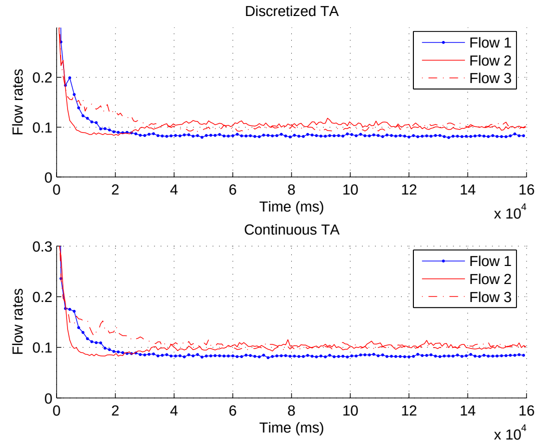


Fig. 5: Flow rates in Network 2 (Grid Topology) with discretized or continuous TA

VIII. Conclusion

In this paper, we have proposed a distributed CSMA scheduling algorithm, and showed that, under the model of perfect CSMA, it is throughput-optimal in wireless networks with a general interference model. We have utilized the product-form stationary distribution of CSMA networks in order to obtain the distributed algorithm and the maximal throughput. Furthermore, we have combined that algorithm with end-to-end flow control to approach the optimal utility, and showed the connection with maximal backpressure scheduling. The algorithm is easy to implement, and the simulation results are encouraging.

The adaptive CSMA algorithm is a modular MAC-layer protocol that can work with other algorithms in the transport layer and network layer. For example, it can be combined with optimal routing, anycast and multicast (Appendix F).

We also considered some practical issues when implementing the algorithm in an 802.11 setting. Since collisions occur in actual 802.11 networks, we discussed a few recent algorithms which explicitly consider collisions and can still approach throughput-optimality.

Our current performance analysis of Algorithm 1 and 2 is based on a separation of time scales, i.e., the vector $\mathbf{r}$ is adapted slowly to allow the CSMA Markov chain to closely track the stationary distribution $\mathbf{p}(\mathbf{r})$. The simulations, however, indicate that such slow adaptations are not always necessary. In the future, we are interested to understand more about the case without time-scale separation.

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*, 2006.

[2] L. Jiang and J. Walrand, "A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks," *the 46th Annual Allerton Conference*, Sep. 23-26, 2008.

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

[4] A. Eryilmaz, A. Ozdaglar and E. Modiano, "Polynomial Complexity Algorithms for Full Utilization of Multi-hop Wireless Networks," *Proceedings of IEEE INFOCOM*, 2007

[5] E. Modiano, D. Shah, and G. Zussman, "Maximizing Throughput in Wireless Networks via Gossiping," *ACM SIGMETRICS Performance Evaluation Review*, vol. 34 , no. 1, June 2006

[6] S. Sanghavi, L. Bui, R. Srikant, "Distributed Link Scheduling with Constant Overhead," *Proceedings of the 2007 ACM SIGMETRICS.*

[7] J. W. Lee, M. Chiang, and R. A. Calderbank, "Utility-Optimal Random-Access Control," *IEEE Transactions on Wireless Communications*, vol. 6, no. 7, pp. 2741-2751, July 2007.

[8] P. Gupta, A.L. Stolyar, "Optimal Throughput Allocation in General Random Access Networks," *Conference on Information Sciences and Systems (CISS'06)*, Princeton, March 2006.

[9] X. Wu, R. Srikant, "Bounds on the Capacity Region of Multi-Hop Wireless Networks under Distributed Greedy Scheduling," *Proceedings of IEEE INFOCOM, 2006.*

[10] X. Lin and N. Shroff, "The impact of imperfect scheduling on cross-layer rate control in multihop wireless networks," *Proceedings of IEEE INFOCOM*, Miami, FL, March 2005.

[11] P. Marbach, A. Eryilmaz and A. Ozdaglar, "Achievable Rate Region of CSMA Schedulers in Wireless Networks with Primary Interference Constraints," *IEEE Conference on Decision and Control*, 2007.

[12] A. Proutiere, Y. Yi, M. Chiang, "Throughput of Random Access without Message Passing," *Conference on Information Sciences and Systems (CISS)*, Princeton, NJ, USA, March 2008.

[13] S. Rajagopalan and D. Shah, "Distributed Algorithm and Reversible Network", *Conference on Information Sciences and Systems (CISS)*, Princeton, NJ, USA, March 2008.

[14] Y. Xi and E. M. Yeh, "Throughput Optimal Distributed Control of Stochastic Wireless Networks," *International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2006.

[15] M. J. Neely, E. Modiano, C. P. Li, "Fairness and Optimal Stochastic Control for Heterogeneous Networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, April 2008, pp. 396-409.

[16] B. Hajek, "Cooling schedules for optimal annealing," *Mathematics of Operations Research*, vol. 13, no. 2, pp. 311–329, 1988.

[17] 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.

[18] 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, December 1992.

[19] N. McKeown, A. Mekkittikul, V. Anantharam, J. Walrand, "Achieving 100% throughput in an input-queued switch," *IEEE Transactions on Communications*, 1999.

[20] F. P. Kelly, "Reversibility and Stochastic Networks," Wiley, 1979.

[21] R. R. Boorstyn, A. Kershenbaum, B. Maglaris, and V. Sahin, "Throughput analysis in multihop CSMA packet radio networks," *IEEE Transactions on Communications*, 35(3):267-274, March, 1987.

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

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

[24] M. Durvy, O. Dousse and P. Thiran, "Border Effects, Fairness, and Phase Transition in Large Wireless Networks", in Proc. *IEEE INFOCOM*, Phoenix, AZ, April 2008.

[25] L. Jiang, S. C. Liew, "Improving Throughput and Fairness by Reducing Exposed and Hidden Nodes in 802.11 Networks," *IEEE Transactions on Mobile Computing*, Vol. 7, No. 1, pp. 34-49, Jan. 2008.

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

[27] L. Jiang, J. Walrand, "Approaching throughput-optimality in a Distributed CSMA Algorithm with Contention Resolution," Technical Report, UC berkeley, Mar. 2009. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-37.html

[28] S. Boyd and L. Vandenberghe, "Convex Optimization", Cambridge University Press, 2004.

[29] P. Whittle, "Systems in stochastic equilibrium," John Wiley & Sons, Inc., 1986, New York, NY, USA.

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

[31] T. Ho and H. Viswanathan, "Dynamic Algorithms for Multicast with Intra-session Network Coding," submitted to *IEEE Transactions on Information Theory*.

[32] R. Ahlswede, N. Cai, S.Y.R. Li, R.W. Yeung, "Network Information Flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204-1216, Jul 2000.

[33] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, 18(3):535–547, 2000.

[34] M. J. Neely and R. Urgaonkar, "Cross Layer Adaptive Control for Wireless Mesh Networks," *Ad Hoc Networks (Elsevier)*, vol. 5, no. 6, pp. 719-743, August 2007.

[35] 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

[36] A. Warrier, S. Ha, P. Wason and I. Rhee, "DiffQ: Differential Backlog Congestion Control for Wireless Multi-hop Networks," Technical Report, Dept. Computer Science, North Carolina State University, 2008.

[37] Loc Bui, R. Srikant, and Alexander Stolyar, "Novel Architectures and Algorithms for Delay Reduction in Back-Pressure Scheduling and Routing," to appear in the *IEEE INFOCOM 2009 Mini-Conference*.

IX. APPENDICES

A. Proof of the fact that $\mathcal{C}$ is the interior of $\bar{\mathcal{C}}$

Definition: The interior of $\bar{\mathcal{C}}$ is defined as int $\bar{\mathcal{C}} := \{\lambda \in \bar{\mathcal{C}} | \mathcal{B}(\lambda, d) \subseteq \bar{\mathcal{C}}$ for some $d > 0\}$, where $\mathcal{B}(\lambda, d) = \{\lambda' \,|\, ||\lambda' - \lambda||_2 \le d\}$.

Proposition 6: λ is strictly feasible if and only if $\lambda \in$ int $\bar{\mathcal{C}}$. (In other words, $\mathcal{C} =$ int $\bar{\mathcal{C}}$.)

Proof: (i) If λ is strictly feasible, then it can be written as $\lambda = \sum_i \bar{p}_i x^i$ where $\bar{p}_i > 0, \forall i$ and $\sum_i \bar{p}_i = 1$. Let $\bar{p}_0$ be the probability corresponding to the all-0 IS, and $\bar{p}_k$ be the probability of the IS $\mathbf{e}_k$, $k = 1, 2, \ldots, K$. Let $d_0 = \min\{\bar{p}_0/K, \min_k \bar{p}_k\} > 0$. We claim that for any λ' that satisfies

$$|\lambda'_k - \lambda_k| \le d_0, \forall k, \tag{25}$$

we have $\lambda' \in \bar{\mathcal{C}}$. Indeed, if λ' satisfies (25), we can find another probability distribution $\bar{\mathbf{p}}'$ such that $\sum_i \bar{p}'_i x^i_k = \lambda'_k, \forall k$. $\bar{\mathbf{p}}'$ can be constructed as follows: let $\bar{p}'_0 = \bar{p}_0 - \sum_k (\lambda'_k - \lambda_k)$, $\bar{p}'_k = \bar{p}_k + (\lambda'_k - \lambda_k)$, and let the probabilites of all other IS's be the same as those in $\bar{\mathbf{p}}$. By condition (25), we have $\bar{\mathbf{p}}' \ge \mathbf{0}$. Also, $\sum_i \bar{p}'_i x^i_k = \lambda'_k, \forall k$.

Therefore, $\mathcal{B}(\lambda, d_0) \subseteq \bar{\mathcal{C}}$ where $d_0 > 0$. So $\lambda \in$ int $\bar{\mathcal{C}}$.

(ii) Assume that $\lambda \in$ int $\bar{\mathcal{C}}$. We now construct a $\mathbf{p} > \mathbf{0}$ such that $\lambda = \sum_i p_i x^i$. First, choose an arbitrary $\mathbf{p}_I > \mathbf{0}$ (such that $\sum_i p_{I,i} = 1$) and let $\lambda_I := \sum_i p_{I,i} x^i$. If it happens to be that $\lambda_I = \lambda$, then λ is strictly feasible. In the following we assume that $\lambda_I \ne \lambda$. Since $\lambda \in$ int $\bar{\mathcal{C}}$, there exists a small-enough $d > 0$ such that $\lambda_{II} := \lambda + d \cdot (\lambda - \lambda_I) \in \bar{\mathcal{C}}$. So λ_{II} can be written as $\lambda_{II} = \sum_i p_{II,i} x^i$ where $\mathbf{p}_{II} \ge \mathbf{0}$ and $\sum_i p_{II,i} = 1$.

Notice that $\lambda = \alpha \cdot \lambda_I + (1-\alpha) \cdot \lambda_{II}$ where $\alpha := d/(1+d) \in (0,1)$. So $\lambda = \sum_i p_i x^i$ where $p_i := \alpha \cdot p_{I,i} + (1-\alpha) \cdot p_{II,i}, \forall i$. Since $\alpha > 0, 1-\alpha > 0$ and $p_{I,i} > 0, p_{II,i} \ge 0, \forall i$, we have $p_i > 0, \forall i$. Therefore λ is strictly feasible. ■

B. Proof the Proposition 2

Consider the convex optimization problem (11), where λ is strictly feasible (i.e., $\lambda = \sum_i \bar{p}_i \cdot x^i$ for some $\bar{p}_i > 0, \forall x^i$ and $\sum_i \bar{p}_i = 1$). Problem (11) is clearly feasible and the feasible region is closed and convex. The objective function (the entropy) is bounded in the feasible region. So, the optimal value is bounded.

We now check whether the Slater condition [28] (pages 226-227) is satisfied. Since all the constraints in (11) are linear, we only need to check whether there exists a *feasible* $\mathbf{u}$ which is in the relative interior [28] of the domain $\mathcal{D}$ of the objective function $-\sum_i u_i \log(u_i)$, which is $\mathcal{D} = \{\mathbf{u} | u_i \geq 0, \sum_i u_i = 1\}$. Since $\lambda = \sum_i \bar{p}_i \cdot x^i$ where $\bar{p}_i > 0, \forall i$ and $\sum_i \bar{p}_i = 1$, letting $\mathbf{u} = \bar{\mathbf{p}}$ satisfies the requirement. Therefore the Slater condition is satisfied. As a result, there exist finite dual variables $y_k^* \geq 0, w_i^* \geq 0, z^*$ such that the Lagrangian

$$\begin{aligned} & \mathcal{L}(\mathbf{u}; \mathbf{y}^*, \mathbf{w}^*, z^*) \\ = \; & -\textstyle\sum_i u_i \log(u_i) + \sum_k y_k^* (\sum_i u_i \cdot x_k^i - \lambda_k) \\ & + z^* (\textstyle\sum_i u_i - 1) + \sum_i w_i^* u_i \end{aligned} \quad (26)$$

is maximized by the optimal solution $\mathbf{u}^*$, and the maximum is attained.

We first claim that the optimal solution satisfies $u_i^* > 0, \forall i$. Suppose $u_i^* = 0$ for all i's in a non-empty set $\mathcal{I}$. For convenience, denote $\bar{\mathbf{p}}$ as the vector of $\bar{p}_i$'s. Since both $\mathbf{u}^*$ and $\bar{\mathbf{p}}$ are feasible for problem (11), any point on the line segment between them is also feasible. Then, if we slightly move $\mathbf{u}$ from $\mathbf{u}^*$ along the direction of $\bar{\mathbf{p}} - \mathbf{u}^*$, the change of the objective function $h(\mathbf{u}) := -\sum_i u_i \log(u_i)$ (at $\mathbf{u}^*$) is proportional to

$$\begin{aligned} & (\bar{\mathbf{p}} - \mathbf{u}^*)^T \nabla h(\mathbf{u}^*) \\ = \; & \sum_i (\bar{p}_i - u_i^*)[-\log(u_i^*) - 1] \\ = \; & \sum_{i \notin \mathcal{I}} (\bar{p}_i - u_i^*)[-\log(u_i^*) - 1] + \sum_{i \in \mathcal{I}} \bar{p}_i [-\log(u_i^*) - 1]. \end{aligned}$$

For $i \notin \mathcal{I}$, $u_i^* > 0$, so $\sum_{i \notin \mathcal{I}} (\bar{p}_i - u_i^*)[-\log(u_i^*) - 1]$ is bounded. But for $i \in \mathcal{I}$, $u_i^* = 0$, so that $-\log(u_i^*) - 1 = +\infty$. Also, since $\bar{p}_i > 0$, we have $(\bar{\mathbf{p}} - \mathbf{u}^*)^T \nabla h(\mathbf{u}^*) = +\infty$. This means that $h(\mathbf{u})$ increases when we slightly move $\mathbf{u}$ away from $\mathbf{u}^*$ towards $\bar{\mathbf{p}}$. Thus, $\mathbf{u}^*$ is not the optimal solution.

Therefore $u_i^* > 0, \forall i$. By complementary slackness, $w_i^* = 0$. So the term $\sum_i w_i^* u_i$ in (26) is 0. Since $\mathbf{u}^*$ maximizes $\mathcal{L}(\mathbf{u}; \mathbf{y}^*, \mathbf{w}^*, z^*)$, it follows that

$$\frac{\partial \mathcal{L}(\mathbf{u}^*; \mathbf{y}^*, \mathbf{w}^*, z^*)}{\partial u_i} = -\log(u_i^*) - 1 + \sum_k y_k^* x_k^i + z = 0, \forall i.$$

Combining these identities and $\sum_i u_i^* = 1$, we have

$$u_i^* = \frac{\exp(\sum_k y_k^* x_k^i)}{\sum_j \exp(\sum_k y_k^* x_k^j)}, \forall i. \quad (27)$$

Plugging (27) back into (26), we have $\max_{\mathbf{u}} \mathcal{L}(\mathbf{u}; \mathbf{y}^*, \mathbf{w}^*, z^*) = -F(\mathbf{y}^*; \lambda)$. Since $\mathbf{u}^*$ and the dual variables $\mathbf{y}^*$ solves (11), $\mathbf{y}^*$ is the solution of $\min_{\mathbf{y} \geq \mathbf{0}} \{-F(\mathbf{y}; \lambda)\}$ (and the optimum is attained). So, $\sup_{\mathbf{r} \geq 0} F(\mathbf{r}; \lambda)$ is attained by $\mathbf{r} = \mathbf{y}^*$. The above proof also shows that (4) is the dual problem of (11).

C. Convergence and stability properties of Algorithm (9)

The following are some main results in [35], which includes the detailed proofs.

(i). Let $r_{max} = +\infty$, i.e., we impose no upper bound on $r_k(j)$. If $\{\alpha(j)\}$ and $\{T(j)\}$ meet certain conditions (which are satisfied by $\alpha(j) = 1/[(j+2)\log(j+2)]$ and $T(j) = j+2$), and $h(r_k(j)) = \min\{c/r_k(j), \bar{w}\}$ where $c, \bar{w} > 0$ (see section V for an explanation of the function), then for any strictly feasible $\lambda \in \mathcal{C}$, $\mathbf{r}(j)$ converges with probability 1 to some $\mathbf{r}^{**}$ which satisfies $s_k(\mathbf{r}^{**}) > \lambda_k, \forall k$. It then follows that the queues are "rate-stable" [35]. (With time-varying $\alpha(j), T(j)$, the system is not time-homogeneous, in which case the "positive Harris recurrence" of system Markov chain is not usually defined. Therefore we use the notion of "rate-stable" here.)

(ii) Let $r_{max} < +\infty$ and $h(r_k(j)) = \epsilon > 0$. Define the capacity region

$$\begin{aligned} \mathcal{C}'(r_{max}, \epsilon) : \; = \; & \{\lambda | \lambda + \epsilon \cdot \mathbf{1} \in \mathcal{C}, \text{ and} \\ & \arg\max_{\mathbf{r} \geq \mathbf{0}} F(\mathbf{r}; \lambda + \epsilon \cdot \mathbf{1}) \in [0, r_{max}]^K\} \end{aligned}$$

If $\lambda \in \mathcal{C}'(r_{max}, \epsilon)$, then there exist constant step size $\alpha(j) = \alpha$ and adjustment period $T(j) = T$ such that all queues are stable. Note that $\mathcal{C}'(r_{max}, \epsilon) \to \mathcal{C}$ as $r_{max} \to +\infty$ and $\epsilon \to 0$. So the maximal throughput can be arbitrarily approximated.

(iii) The case with $h(r_k(j)) = 0$ (i.e., Algorithm 1): Similar to (i), for any $\lambda \in \mathcal{C}$, with $r_{max} = +\infty$ and $\alpha(j), T(j)$ in (i), $\mathbf{r}(j)$ converges with probability 1 to $\mathbf{r}^*$, the solution of (4), which satisfies $s_k(\mathbf{r}^*) \geq \lambda_k, \forall k$, and the queues are rate-stable. Similar to (ii), with $r_{max} < +\infty$ and assume that $\lambda \in \mathcal{C}'(r_{max}, 0)$, then one can choose constant $\alpha(j) = \alpha$ and $T(j) = T$ such that the long-term average service rates are arbitrarily close to the arrival rates. (As $r_{max} \to \infty$, $\mathcal{C}'(r_{max}, 0) \to \mathcal{C}$.)

D. Proof of Proposition 4

Since in the optimal solution of problem (14), $\mathbf{f} = \hat{\mathbf{f}}$ and $\mathbf{u} = \hat{\mathbf{u}}$. we have

$$-\sum_i \hat{u}_i \log(\hat{u}_i) + \beta \sum_{m=1}^M v_m(\hat{f}_m) \geq -\sum_i \bar{u}_i \log(\bar{u}_i) + \beta \cdot \bar{W}.$$

Therefore,

$$\beta[\sum_{m=1}^M v_m(\hat{f}_m) - \bar{W}] \geq -\sum_i \bar{u}_i \log(\bar{u}_i) + \sum_i \hat{u}_i \log(\hat{u}_i).$$

Notice that $|-\sum_i \bar{u}_i \log(\bar{u}_i) + \sum_i \hat{u}_i \log(\hat{u}_i)| \leq K \cdot \log 2$, so

$$\beta[\sum_{m=1}^M v_m(\hat{f}_m) - \bar{W}] \geq -K \cdot \log 2.$$

Also, clearly $\bar{W} \geq \sum_{m=1}^M v_m(\hat{f}_m)$, so

$$-\frac{K \cdot \log 2}{\beta} \leq \sum_{m=1}^M v_m(\hat{f}_m) - \bar{W} \leq 0. \quad (28)$$

E. Analysis of Algorithm 2

Regard each period (with length T) as a "time slot" in [15]. Using the proof of Proposition 5, $b_{km}(j) \leq \beta \cdot V + \alpha + 2\alpha \cdot (L-1), \forall k, m, j$. Since $r_k(j) = [\max_m b_{km}(j)]_+$, we have $0 \leq r_k(j) \leq C := \beta \cdot V + \alpha + 2\alpha \cdot (L-1)$. Thus, the mixing time of the CSMA Markov chain in any period is bounded [35]. So

$$|E_j[s'_k(j)] - s_k(\mathbf{r}(j))| \leq \frac{C_1}{T} \quad (29)$$

where the constant C_1 depends on C and K ([35]), and $E_j(\cdot)$ means the expectation conditioned on the values of all random variables up to time t_j.

Since $u_i^* := p_i(\mathbf{r}(j)), \forall i$ maximizes $H(\mathbf{u}) + \sum_k [r_k(j) \sum_i (x_k^i \cdot u_i)]$ (see Proposition 3), similar to the proof of Proposition 4, we have

$$\begin{aligned} & \sum_k [r_k(j) \sum_i (x_k^i \cdot u_i^*)] \\ = & \sum_k [r_k(j) \cdot s_k(\mathbf{r}(j))] \\ \geq & \max_{\mu \in \bar{\mathcal{C}}} \sum_k [r_k(j) \cdot \mu_k(\mathbf{r}(j))] - K \cdot \log(2) \end{aligned}$$

where $\bar{\mathcal{C}}$ is the set of feasible service rates (including $\mathcal{C}$ and its boundary).

By this inequality and (29),

$$\begin{aligned} \sum_k \{r_k(j) \cdot E_j[s'_k(j)]\} \geq & \max_{\mu \in \bar{\mathcal{C}}} \sum_k [r_k(j) \cdot \mu_k(\mathbf{r}(j))] \\ & -K \cdot \log(2) - K \cdot C \cdot C_1/T. \end{aligned}$$

Define $\tilde{r}_k(j) := r_k(j)/\alpha$ (then $\tilde{r}_k(j)$ corresponds to the maximal differential backlog $W_k^*(j)$ in [15], since the change of $r_k(j)$ has been scaled by the step size α), we have

$$\begin{aligned} \sum_k \{\tilde{r}_k(j) \cdot E_j[s'_k(j)]\} \geq & \max_{\mu \in \bar{\mathcal{C}}} \sum_k [\tilde{r}_k(j) \cdot \mu_k(\mathbf{r}(j))] \\ & -[K \cdot \log(2) + K \cdot C \cdot C_1/T]/\alpha. \end{aligned}$$

Now, using Corollary 1 in [15], it follows that

$$\begin{aligned} & \liminf_{J \to \infty} \sum_m v_m(\bar{f}_m(J)) \\ \geq & \bar{W} - \frac{2[K \cdot \log(2) + K \cdot C \cdot C_1/T]/\alpha + 5K}{2\beta/\alpha} \\ = & \bar{W} - \frac{[K \cdot \log(2) + K \cdot C \cdot C_1/T] + 5\alpha \cdot K/2}{\beta} \quad (30) \end{aligned}$$

where $\bar{f}_m(J) := \sum_{j=0}^{J-1} E[f_m(j)]/J$ is the expected average rate of flow m up to the $J-1$'s period. We have used the fact that $R_k^{max} = 1$, $\mu_{max,k}^{in} = \mu_{max,k}^{out} = 1$, where R_k^{max} is the maximal flow input rate at link k, $\mu_{max,k}^{in}$ and $\mu_{max,k}^{out}$ are the maximal rate the link k can receive or transmit.

As expected, when $T \to \infty$ and $\alpha \to 0$, this bound matches the bound in Proposition 4. Also, as $\beta \to \infty$, $\alpha \to 0$, and $T \to \infty$ in a proper way (since C and C_1 depend on β), $\liminf_{J \to \infty} \sum_m v_m(\bar{f}_m(J)) \to \bar{W}$.

The above bound (30), however, is not very tight. Our simulation shows good performance without a very large β, T or a very small α.

F. Extensions: Adaptive CSMA scheduling as a modular MAC-layer protocol

Using derivations similar to section IV-A, our CSMA algorithm can serve as a modular "MAC-layer scheduling component" in cross-layer optimization, combined with other components in the transport layer and network layer, with queue lengths as the shared information. For example, we demonstrate in this section its combination with optimal multipath routing, multi-channel selection, anycast, and multicast.

1) Anycast: To make the formulation more general, let's consider anycast with multipath routing. (This include unicast with multipath routing as a special case.) Assume that there are M flows. Each flow m has a source $\delta(m)$ (with some abuse of notation) which generates data and a set of destinations $\mathcal{D}(m)$ which receive the data. "Anycast" means that it is sufficient for the data to reach any node in the set $\mathcal{D}(m)$. However, there is no specific "path" for each flow. The data generated by the source is allowed to split and traverse any link before reaching the destinations (i.e., multipath routing). This allows better utilization of the network resource by routing the data through less congested parts of the network. (For simplicity, we don't consider the possibility of physical-layer multicast here, i.e., the effect that a node's transmission can be received by multiple nodes simultaneously.)

In this case, it is more convenient to use a "node-based" formulation [1], [14]. Denote the number of nodes by J. For each node j, let $\mathcal{I}(j) := \{k | (k,j) \in \mathcal{L}\}$, where $\mathcal{L}$ is the set of links (it is also the set $\mathcal{V}$ in the link contention graph), and let $\mathcal{O}(j) := \{k | (j,k) \in \mathcal{L}\}$. Denote the rate of flow m on link (j,l) by s_{jl}^m. Then the (approximate) utility maximization problem, similar to (14), is

$$\begin{aligned} \max_{\mathbf{u},\mathbf{s},\mathbf{f}} \quad & -\textstyle\sum_i u_i \log(u_i) + \beta \cdot \sum_{m=1}^M v_m(f_m) \\ \text{s.t.} \quad & s_{jl}^m \geq 0, \forall (j,l) \in \mathcal{L}, \forall m \\ & f_m + \textstyle\sum_{l \in \mathcal{I}(j)} s_{lj}^m \leq \sum_{l \in \mathcal{O}(j)} s_{jl}^m, \forall m, j = \delta(m) \\ & \textstyle\sum_{l \in \mathcal{I}(j)} s_{lj}^m \leq \sum_{l \in \mathcal{O}(j)} s_{jl}^m, \forall m, \\ & \quad j \neq \delta(m), j \notin \mathcal{D}(m) \\ & \textstyle\sum_i u_i \cdot x_{(j,l)}^i = \sum_m s_{jl}^m, \forall (j,l) \in \mathcal{L} \\ & u_i \geq 0, \textstyle\sum_i u_i = 1. \end{aligned}$$

Associate a dual variable $q_j^m \geq 0$ to the 2nd and 3rd lines of constraints (for each m and $j \notin \mathcal{D}(m)$), and define $q_j^m = 0$ if $j \in \mathcal{D}(m)$. (Note that there is no flow-conservation constraint for flow m at each node in $\mathcal{D}(m)$.) Then similar to section IV-A, a partial Lagrangian is

$$\begin{aligned} & \mathcal{L}(\mathbf{u}, \mathbf{s}, \mathbf{f}; \mathbf{q}) \\ = & -\textstyle\sum_i u_i \log(u_i) \\ & +\beta \cdot \textstyle\sum_m v_m(f_m) - \sum_m q_{\delta(m)}^m f_m \\ & + \textstyle\sum_{(j.l) \in \mathcal{L}, m} s_{jl}^m [(q_j^m - q_l^m)]. \end{aligned} \quad (31)$$

First fix $\mathbf{u}$ and $\mathbf{q}$, consider maximizing $\mathcal{L}(\mathbf{u}, \mathbf{s}, \mathbf{f}; \mathbf{q})$ over $\mathbf{s}$, subject to $s_{jl}^m \geq 0$ and $\sum_i u_i \cdot x_{(j,l)}^i = \sum_m s_{jl}^m$. Clearly, for each link (j,l), the flow with the maximal back-pressure $z_{jl} := \max_m (q_j^m - q_l^m)$ should be served (with the whole rate $\sum_i u_i \cdot x_{(j,l)}^i$). Plug this solution of $\mathbf{s}$ back to (31), the rest derivation is the same as in section IV-A. Therefore the distributed algorithm is as follows.

Initially, assume that all queues are empty, and set $q_j^m = 0, \forall j, m$. Then iterate:

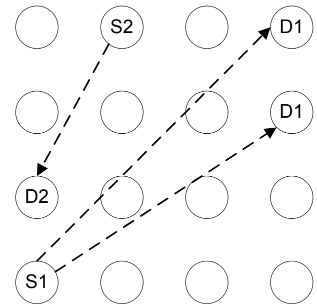
- Link (j,l) chooses to serve a flow with the maximal back-pressure $z_{(j,l)} = \max_m (q_j^m - q_l^m)$ when it gets the opportunity to transmit. (Note that there is no replication of packets.)
- Link (j,l) lets $r_{(j,l)} = z_{(j,l)}$ in the CSMA operation.
- Rate control: For each flow m, if node j is its source, then it sets $f_m = \arg\max_{f_m'} \{\beta \cdot v_m(f_m') - q_j^m f_m'\}$.
- The dual variables q_j^m are updated by a sub-gradient algorithm: $q_j^m \leftarrow [q_j^m + \alpha(\sum_{l\in\mathcal{I}(j)} s_{lj}^m - \sum_{l\in\mathcal{O}(j)} s_{jl}^m)]_+$ if $j \neq \delta(m)$ and $j \notin \mathcal{D}(m)$; and $q_j^m \leftarrow [q_j^m + \alpha(f_m + \sum_{l\in\mathcal{I}(j)} s_{lj}^m - \sum_{l\in\mathcal{O}(j)} s_{jl}^m)]_+$ if $j = \delta(m)$. (By doing this, $q_j^m \propto Q_j^m$ where Q_j^m is the corresponding queue length.) Always let $q_j^m = 0$ if $j \in \mathcal{D}(m)$.

Furthermore, the above algorithm can be readily extended to channel selection in *multi-channel wireless networks*, with each "link" defined by a triplet $(j,l;c)$, which refers to the logical link from node j to l on channel c. In this scenario, the link contention graph is defined on the set of links $(j,l;c)$.

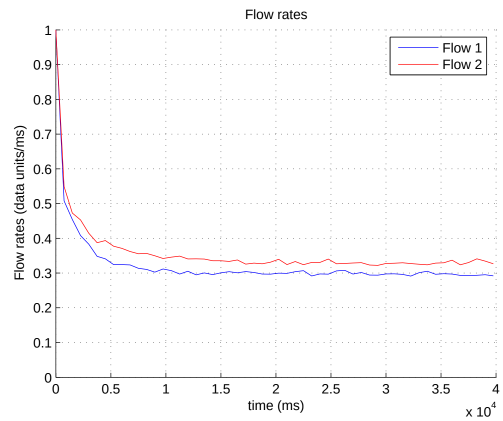
To give a numerical example (with a single channel), we simulate two flows in a grid topology in Fig. 6 (a). The transmission range and the interference relationships are the same as the grid network in Fig. 4 (a). For flow 1, the source node is S1 and the two destination nodes are labeled by D1. For flow 2, the source node is S2, and the destination node is D2. (Note that the dashed lines only mean the abstract directions but not the real paths of the flows.) Let $v_m(\cdot) = \log(\cdot), m = 1,2$, and $\beta = 3$. The resulting flow rates are shown in Fig. 6 (b).

2) Multicast with network coding: Assume that there are M multicast sessions. Each session m has a source $\delta(m)$ which generates data and a set of destinations $\mathcal{D}(m)$ which receive the data. Different from "anycast", here the data must reach all nodes in the set $\mathcal{D}(m)$. There are two possible designs for multicast. (1) Fixed multicast tree, where the routes of each multicast session is fixed. (2) Multicast combined with multipath routing and network coding. Case (1) is straightforward, but the routing may not be optimal. In case (2), [31] demonstrates an algorithm which achieves the optimal utility, which however, requires *centralized* Maximal-Weight scheduling at the MAC layer. In this section, we show that CSMA scheduling can be combined with it, leading to a *fully distributed* algorithm. To facilitate network coding, we let all the packets have the same size (Note that our results are *insensitive* to the distribution of the transmission time, i.e., packet size, if the transmission time and waiting time are not both constant [23]).

According to the theory of network coding [32], a certain flow rate for a multicast session can be supported if and only if it can be supported separately for each destination node. Let s_{jl}^{mp} be the information flow rate on link (j,l) in multicast session m destined for node $p \in \mathcal{D}(m)$, and s_{jl}^m be the "capacity" for session m on link (j,l). The above condition is that $s_{jl}^{mp} \le s_{jl}^m, \forall p \in \mathcal{D}(m)$. Then, the approximate utility



(a) Network topology (S1, S2 are sources; D1, D2 are destinations)



(b) Flow rates

Fig. 6: Anycast with multipath routing

maximization problem is

$$\begin{array}{ll} \max_{\mathbf{u},\mathbf{s},\mathbf{f}} & H(\mathbf{u}) + \beta \cdot \sum_{m=1}^{M} v_m(f_m) \\ \text{s.t.} & s_{jl}^{mp} \ge 0, \forall (j,l) \in \mathcal{L}, \forall m, \forall p \in \mathcal{D}(m) \\ & f_m + \sum_{l\in\mathcal{I}(j)} s_{lj}^{mp} \le \sum_{l\in\mathcal{O}(j)} s_{jl}^{mp}, \\ & \forall m, j = \delta(m), p \in \mathcal{D}(m) \\ & \sum_{l\in\mathcal{I}(j)} s_{lj}^{mp} \le \sum_{l\in\mathcal{O}(j)} s_{jl}^{mp}, \\ & \forall m, p \in \mathcal{D}(m), j \neq \delta(m), j \neq p \\ & s_{jl}^{mp} \le s_{jl}^m, \forall p \in \mathcal{D}(m), \forall (j,l) \in \mathcal{L} \\ & \sum_i u_i \cdot x_{(j,l)}^i = \sum_m s_{jl}^m, \forall (j,l) \in \mathcal{L} \\ & u_i \ge 0, \sum_i u_i = 1. \end{array}$$

Associate a dual variable $q_j^{mp} \ge 0$ to the 2nd and 3rd lines of constraints (for each $m, p \in \mathcal{D}(m)$ and $j \neq p$), and define $q_j^{mp} = 0$ if $j = p$. Then a partial Lagrangian is

$$\begin{aligned} & \mathcal{L}(\mathbf{u},\mathbf{s},\mathbf{f};\mathbf{q}) \\ = \; & H(\mathbf{u}) \\ & + \beta \cdot \sum_m v_m(f_m) - \sum_m (\sum_{p\in\mathcal{D}(m)} q_{\delta(m)}^{mp}) f_m \\ & + \sum_{(j.l)\in\mathcal{L}, m, p\in\mathcal{D}(m)} s_{jl}^{mp} [(q_j^{mp} - q_l^{mp})]. \end{aligned} \tag{32}$$

We first optimize $\mathcal{L}(\mathbf{u},\mathbf{s},\mathbf{f};\mathbf{q})$ over $\{s_{jl}^{mp}\}$, subject to $0 \le s_{jl}^{mp} \le s_{jl}^m$. A solution is as follows: $s_{jl}^{mp} = 0, \forall p$ satisfying $q_j^{mp} - q_l^{mp} \le 0$, and $s_{jl}^{mp} = s_{jl}^m, \forall p$ satisfying $q_j^{mp} - q_l^{mp} > 0$. Define the "back-pressure" of session m on link (j,l) as $W_{jl}^m := \sum_{p\in\mathcal{D}(m)} (q_j^{mp} - q_l^{mp})_+$.

By plugging the above solution to (32), we have

$$\begin{aligned} & \mathcal{L}(\mathbf{u},\mathbf{s},\mathbf{f};\mathbf{q}) \\ = \; & H(\mathbf{u}) \\ & +\beta\cdot\sum_m v_m(f_m) - \sum_m\Big(\sum_{p\in\mathcal{D}(m)} q^{mp}_{\delta(m)}\Big)f_m \\ & +\sum_{(j.l)\in\mathcal{L},m} s^m_{jl}W^m_{jl}. \end{aligned} \qquad (33)$$

Now we optimize it over $\{s^m_{jl}\}$, subject to $\sum_i u_i \cdot x^i_{(j,l)} = \sum_m s^m_{jl}$. One can find that the following is similar to previous derivations. To void repetition, we directly write down the algorithm.

Initially, assume that all queues are empty, and set $q^{mp}_j = 0, \forall j, m, p$. Then iterate:

- Link (j,l) chooses to serve a session m' with the maximal back-pressure $z_{(j,l)} := \max_m W^m_{jl}$ when it gets the opportunity to transmit, where $W^m_{jl} := \sum_{p\in\mathcal{D}(m)}(q^{mp}_j - q^{mp}_l)_+$. To serve session m', node j performs a random linear combination[8] of the head-of-line packets from the queues of session m' with destination $p \in \mathcal{D}(m')$ which satisfies $q^{m'p}_j - q^{m'p}_l > 0$, and transmits the coded packet (similar to [31]). The coded packet, after received by node l, is replicated and put into corresponding queues of session m' at node l (with destination $p \in \mathcal{D}(m')$ such that $q^{m'p}_j - q^{m'p}_l > 0$). The destinations can eventually decode the source packets [31].
- Link (j,l) lets $r_{(j,l)} = z_{(j,l)}$ in the CSMA operation.
- Rate control: For each flow m, if node j is its source, then it sets $f_m = \arg\max_{f'_m}\{\beta \cdot v_m(f'_m) - (\sum_{p\in\mathcal{D}(m)} q^{mp}_{\delta(m)})f'_m\}$.
- The dual variables q^m_j are updated by a sub-gradient algorithm: $q^{mp}_j \leftarrow [q^{mp}_j + \alpha(\sum_{l\in\mathcal{I}(j)} s^{mp}_{lj} - \sum_{l\in\mathcal{O}(j)} s^{mp}_{jl})]_+$ if $j \neq \delta(m)$ and $j \neq p$ where $p \in \mathcal{D}(m)$; and $q^{mp}_j \leftarrow [q^{mp}_j + \alpha(f_m + \sum_{l\in\mathcal{I}(j)} s^{mp}_{lj} - \sum_{l\in\mathcal{O}(j)} s^{mp}_{jl})]_+$ if $j = \delta(m)$. (Note that each packet generated by the source $j = \delta(m)$ is replicated and enters the queues at the source for all destinations of session m.) By doing this, $q^{mp}_j \propto Q^{mp}_j$ where Q^{mp}_j is the corresponding queue length. Always let $q^{mp}_j = 0$ if $j = p$ where $p \in \mathcal{D}(m)$.

We simulate the same topology as in the "anycast" case, where the two flows (or "sessions") have the same sources and destinations as before. The difference is that in the first session we have to send the data to both of its destinations. Let $v_m(\cdot) = \log(\cdot), m = 1, 2$, and $\beta = 3$. The result is shown in Fig. 7.

[8]We briefly explain how to perform a "random linear combination" of these packets. For more details, please refer to [31]. (Note that *our main focus* here is to show how to combine CSMA scheduling with other network protocols, instead of network coding itself.) Initially, each packet generated by the source in each session is associated with an ID. Assume that each packet is composed of many "blocks", where each block has γ bits. So, each block can be viewed as a number in a finite field F_{2^γ} which has 2^γ elements. For each packet P to be combined here, randomly choose a coefficient $a_P \in F_{2^\gamma}$. Denote the i'th block of packet P as $P(i)$. Then the corresponding block in the code packet Z is computed as $Z(i) = \sum_P a_P P(i)$, where the multiplication and summation is on the field F_{2^γ}, and the summation is over all the packets to be combined.

Clearly, each packet in the network is a linear combination of some source packets. The ID's of these source packets and the corresponding coefficients are included in the packet header, and are updated after each linear combination along the path (such that the destinations can decode the source packets).

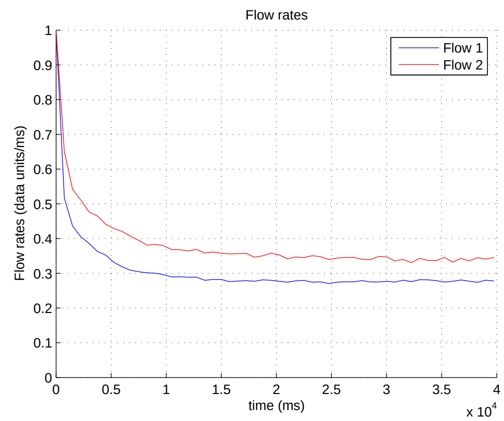


Fig. 7: Multicast with multipath routing and network coding

PLACE PHOTO HERE

Libin Jiang received his B.Eng. degree in Electronic Engineering & Information Science from the University of Science and Technology of China in 2003 and the M.Phil. degree in Information Engineering from the Chinese University of Hong Kong in 2005, and is currently working toward the Ph.D. degree in the Department of Electrical Engineering & Computer Science, University of California, Berkeley. His research interest includes wireless networks, communications and game theory.

PLACE PHOTO HERE

Jean Walrand received his Ph.D. in EECS from UC Berkeley, where he has been a professor since 1982. He is the author of An Introduction to Queueing Networks (Prentice Hall, 1988) and of Communication Networks: A First Course (2nd ed. McGraw-Hill,1998) and co-author of High Performance Communication Networks (2nd ed, Morgan Kaufman, 2000). Prof. Walrand is a Fellow of the Belgian American Education Foundation and of the IEEE and a recipient of the Lanchester Prize and of the Stephen O. Rice Prize.