Joint Congestion Control and Distributed Scheduling for Throughput Guarantees in Wireless Networks

Gaurav Sharma
School of Elec. & Comp. Engg.
Purdue University
West Lafayette, IN 47907
gsharma@purdue.edu

Ness B. Shroff
School of Elec. & Comp. Engg.
Purdue University
West Lafayette, IN 47907
shroff@purdue.edu

Ravi R. Mazumdar
Dept. of Elec. & Comp. Engg.
Univ. of Waterloo
Waterloo, ON N2L 3G1
mazum@ece.uwaterloo.ca

***Abstract*— We consider the problem of throughput-optimal cross-layer design of wireless networks. We propose a joint congestion control and scheduling algorithm that achieves a fraction $d_I(G)$ of the capacity region, where $d_I(G)$ depends on certain structural properties of the underlying connectivity graph G of the wireless network and also on the type of interference constraints. For a wide range of wireless networks, $d_I(G)$ can be upper bounded by a constant, independent of the number of nodes in the network. The scheduling element of our algorithm is the maximal scheduling policy. Although maximal scheduling policy has been considered in many of the previous works, the difficulties that arise in implementing it in a distributed fashion in the presence of interference have not been dealt with previously. In this paper, we propose two novel randomized distributed algorithms for implementing the maximal scheduling policy under the 1-hop and 2-hop interference models.**

I. Introduction

Wireless networks have become a ubiquitous part of all modern day communication systems. Unlike wireline networks, where bandwidth and other resources are plentiful, wireless networks are highly resource constrained, thus underscoring the need for efficient utilization of the wireless resources. A seminal contribution in this direction was made in [27], where the authors characterized the *capacity region* of constrained queuing systems, such as a wireless network. They also developed a queue length based scheduling scheme that is *throughput-optimal*, i.e., it stabilizes the network provided the user rates fall within the capacity region of the network, where the capacity region is defined to be the set of user arrival rates under which the network is stable (the queue lengths at all the nodes are bounded).

Unlike wireline networks, where all links have fixed capacities, the capacity of a wireless link varies with channel variations due to fading; changes in power allocation, link scheduling, or routing; and changes in network topology, etc. This results in the capacity region of a wireless network having a joint dependence on routing, power allocation, link scheduling, and channel variations. In order to maximize the capacity region of the network, one must therefore develop algorithms that can jointly optimize routing, link scheduling, and power control under possibly varying channel conditions and network topology. This has spurred recent interest in developing cross-layer optimization algorithms (see, for example, [29], [17], [16], [26], [5]).

Motivated by the works on fair resource allocation in wireline networks [8], [23], [13], [2], [30], researchers have also incorporated congestion control into the cross-layer optimization framework [3], [10], [9], [15], [28], [25], [31], [19]. The congestion control component controls the rates at which users inject data into the network so as to ensure that they fall within the capacity region of the network.

The most important component of any cross-layer optimization algorithm is the scheduler that needs to solve a very difficult global optimization problem of the form:

$$\textbf{maximize} \sum_{l\in\mathcal{L}} p_l r_l \qquad (1)$$
$$\textbf{subject to } \mathbf{r} \in \Delta$$

where $\mathcal{L}$ denotes the set of wireless links; $\mathbf{r}$ is the vector of link rates r_l, $l \in \mathcal{L}$; p_l, $l \in \mathcal{L}$, is the congestion price or possibly some function of the backlog at link l; and Δ is the capacity region of the network.

The main difficulty in solving the above optimization problem is that the capacity region Δ depends on the complete network topology and, in general, has no simple representation in terms of the power constraints at the individual links or nodes. The above optimization problem is, in general, NP-Complete and Non-Approximable*.

The above problem has been studied under several special cases of interest, e.g., with simplified interference models and no power control. The interference models studied in the literature include the node-exclusive interference model [7], [1], [4], [10], [3], [27], [26], [19] and the IEEE 802.11 type interference model [28], [24], [6]. Both these models are specific instances of the class of K-hop interference models studied in our previous work [21], [22]. An interference model is termed a K-hop interference model if the only constraint imposed on the set of simultaneously active links is that no two

*A problem is said to be Non-Approximable if it does not admit any constant factor polynomial time approximation algorithm.

links from the set should be within K hops of each other. By increasing K, one can model more and more stringent interference constraints. The node-exclusive interference model (used for Bluetooth and FH-CDMA networks [14], [1], [7]) and IEEE 802.11 type interference model (used for IEEE 802.11 networks [28], [6]) correspond to 1-hop and 2-hop interference models, respectively.

The optimal scheduling problem is polynomial time solvable under the 1-hop interference model, however, it is NP-Hard and Non-Approximable under all K-hop interference models for $K > 1$ (see [21]). In [22], we showed that the optimal scheduling problem can be approximated within a constant factor under all K-hop interference models for wireless networks whose connectivity graph is a geometric graph. Similar results can be derived for disk graphs and (r, s)-civilized graphs. These results are quite encouraging as a wide variety of wireless networks can be modeled using the above families of graphs.

In this paper, we propose a joint congestion control and scheduling algorithm that provides provable throughput guarantees under all contention matrix based interference models. In particular, for wireless networks whose connectivity graph is a geometric graph the proposed algorithm achieves a constant fraction of the capacity region under all K-hop interference models. The scheduling element of our algorithm is the maximal scheduling policy, which has been studied in many of the previous works [4], [28], [10]. We also provide randomized distributed algorithms for implementing the maximal scheduling policy under 1-hop and 2-hop interference models.

The rest of the paper is organized as follows. We describe our system model and discuss some related work in Section II. An upper bound on the capacity region is derived in Section III. The joint congestion control and scheduling algorithm is developed in Section IV, and a lower bound on its performance is derived. Randomized distributed algorithms for implementing the maximal scheduling policy under 1-hop and 2-hop interference models are proposed in Section V. Finally, concluding remarks are presented in Section VI.

II. System Model and Related Work

We consider a set V of nodes, labeled $1, 2, ..., |V|$, communicating with each other using wireless means. We say that link (u, v) joining node u to node v exists if node u can successfully transmit to node v, provided no other node in the network transmits at the same time. The set of links so formed is denoted by E. Note that the existence of a link between any two nodes depends on many factors (e.g., noise variance at the receiving node, coding and modulation scheme used by the nodes). Although, we do not consider channel variations in this paper, they can easily be incorporated into our model. We refer the interested reader to [17], [16], [11] for related results.

We consider K types of users, labeled $1, 2, ..., K$, sending data over the network. We assume that type k users arrive into the network according to a Poisson process with rate λ_k. Each user of type k, brings with it a file of size $1/\mu_k$ to be transferred over the network. We assume that all users of any given type send their data over the same, loop-free route. The extension to the multi-route case is straightforward; we refer the reader to [17], [16], [11], [26] for related results. The user routes are stored in an incidence matrix $[H_k^l]$, where $H_k^l = 1$ if link l belongs to the route of type k users; and 0 otherwise.

The interference constraints are modeled using a contention matrix $[C_{ij}]_{i,j \in E}$. More precisely, link i is said to interfere with link j if $C_{ij} = 1$; no two links which interfere can be scheduled at the same time. All diagonal entries of C are set to 1. Time is divided into slots of unit duration. Link l can transmit at rate c_l during a slot if no other interfering link is scheduled to transmit during the same slot. Such an interference model has been widely used in the literature (see, for example, [4]), and the interference models used in many other works [21], [22], [10], [9], [11], [4], [28], [3] can be obtained by imposing some additional constraints on the contention matrix. Unless otherwise stated, the only constraint we impose on $[C_{ij}]$ in the paper is that it should be symmetric; i.e., link i interferes with link j if and only if link j interferes with link i.

Let $\vec{\lambda} = (\lambda_1, \lambda_2, ..., \lambda_K)$ be the vector of user arrival rates. Let $n_k(t)$ and $Q_l(t)$ denote the number of type k users and queue backlog at link l in the network at time t, respectively. As in [10], [17], we say that the network is stable if

$$\limsup_{t \to \infty} \frac{1}{t} \int_0^t 1_{\{\sum_{k=1}^K n_k(t) + \sum_{l \in E} Q_l(t) > N\}} dt \to 0 \quad (2)$$

as $N \to \infty$. The capacity region of the network is defined to be the set of user arrival rate vectors for which the network can be stabilized by some scheduling policy. The capacity region of a constrained queuing system, such as a wireless network, is well characterized in [27]. For our model, the capacity region is given by the set

$$\Omega = \left\{ \vec{\lambda} : \left[\sum_{k=1}^K \frac{H_k^l \lambda_k}{\mu_k c_l} \right]_{l \in E} \in Co(\mathcal{S}) \right\}, \quad (3)$$

where $Co(\mathcal{S})$ represents the convex hull of all link schedules $\mathcal{S}$ that satisfy the constraints imposed by our interference model.

A scheduling scheme is said to be *throughput-optimal* if it stabilizes the network for all user arrival rate vectors within Ω. In [27], the authors proposed a throughput-optimal queue length based scheduling scheme. However, their scheme requires centralized computation and is computationally expensive (NP-Hard) in several cases of interest. Since it is difficult to do centralized computation in ad hoc settings, a considerable amount of effort has been put forth in devising simple distributed schemes that can achieve a certain fraction of the capacity region.

A scheduling scheme that has been widely studied in this context is the so-called *maximal scheduling policy* [10], [4],

[22], [28]†. In [10] and [28], the performance of maximal scheduling policy is studied under a joint congestion control and scheduling framework with multi-hop traffic, and 1-hop and 2-hop interference models, respectively. In [4] and [22], a slightly more restrictive setting with single-hop traffic and no congestion control is considered under the contention matrix based interference model and K-hop interference model, respectively.

Although maximal scheduling policy has been considered in many of the previous works, the difficulties that arise in implementing it in a distributed fashion in the presence of interference have not been dealt with previously. Next, we highlight the main contributions of this paper.

- A joint congestion control and scheduling algorithm based on maximal scheduling policy is proposed under a multi-hop setting with contention matrix based interference model and is shown to achieve a fraction $d_I(G)$ of the capacity region, where $d_I(G)$ is the interference degree of underlying connectivity graph G (see Section III). These results extend earlier results in [4], which were derived under a single-hop (MAC layer) setting with no congestion control.
- The performance of the proposed algorithm is shown to be at most a constant factor from the optimal under all K-hop interference models, provided the underlying connectivity graph G is a geometric graph. These results extend our earlier results in [22].
- Two randomized distributed algorithms are proposed for implementing the maximal scheduling policy under 1-hop and 2-hop interference models. Both these algorithms fully account for the link interferences in a wireless setting and require $\Theta(\Delta \log^2 |V|)$ rounds of computation and local message exchange, respectively, where Δ is the maximum node degree in the network.

III. AN UPPER BOUND ON THE CAPACITY REGION

In this section, we derive an upper bound on the capacity region under a contention matrix based interference model. The upper bound depends on the *interference degree* of the network graph, which we define more formally next (see also [4] and [22]).

Definition 1: The interference set $I(e)$ of link e is the set of links that interfere with link e, i.e.,

$$I(e) = \{l \in E : C_{el} = 1\}.$$

Definition 2: A set of links A is said to be a non-interfering set if it does not contain any interfering links, i.e., for each pair of links $u, v \in A$ with $u \neq v$, we have $C_{uv} = 0$.

Definition 3: The interference degree $d_I(e)$ of link e is the maximum number of links belonging to its interference set that do not interfere with each other, i.e.,

$$d_I(e) = \max_{A \subseteq I(e): A \text{ is a non-interfering set}} |A|.$$

†Note that the terminologies used in these works and some minor details of the schemes differ slightly from each other, but the main idea is essentially the same.

Definition 4: The interference degree $d_I(G)$ of graph $G = (V, E)$ is the maximum interference degree across its constituent links, i.e., $d_I(G) = \max_{e \in E} d_I(e)$.

We are now ready to upper bound the capacity region.

Theorem 1: The capacity region Ω specified by (3) consists of user arrival rate vectors $\vec{\lambda}$ that satisfy $\sum_{l \in I(e)} \sum_{k=1}^{K} \frac{H_k^l \lambda_k}{\mu_k c_l} \leq d_I(e)$ for all $e \in E$.

Proof: Let $\mathcal{S}$ be a link schedule that activates the same set of non-interfering links $A_{\mathcal{S}}$ during every slot. Consider a link $e \in E$. Since the set of links $A_{\mathcal{S}}$ is non-interfering, it contains at most $d_I(e)$ links from $I(e)$. Thus, the link rate vectors $[x_l]_{l \in E}$ under $A_{\mathcal{S}}$ must satisfy

$$\sum_{l \in I(e)} \frac{x_l}{c_l} \leq d_I(e) \text{ for all } e \in E. \tag{4}$$

Since this result holds for all link schedules that satisfy the interference constraints, it follows that all *feasible* link rate vectors under the contention matrix based interference model must satisfy the constraints given in (4). The result now follows by noting that a user arrival rate vector $\vec{\lambda}$ induces an average load of $\sum_{k=1}^{K} H_k^l \lambda_k / \mu_k$ on link l. ■

The following result is a direct consequence of Theorem 1 and the fact that $d_I(G) = \max_{e \in E} d_I(e)$.

Corollary 1: Any user arrival rate vector $\vec{\lambda}$ that belongs to the capacity region Ω specified by (3) satisfies $\sum_{l \in I(e)} \sum_{k=1}^{K} \frac{H_k^l \lambda_k}{\mu_k c_l} \leq d_I(G)$ for all $e \in E$.

IV. JOINT CONGESTION CONTROL AND SCHEDULING FOR THROUGHPUT GUARANTEES

A. The Algorithm

We now propose a joint congestion control and scheduling algorithm that is guaranteed to achieve a fraction $d_I(G)$ of the capacity region under any contention matrix based interference model. The algorithm maintains *congestion prices* $q_l(t)$, $l \in E$ to estimate the level of congestion in the network at time t. The congestion control and scheduling are performed using these congestion prices. Time is divided into slots of unit duration, and both congestion prices and user rates are updated at the beginning of each slot. The detailed description of the algorithm follows:

Algorithm:CCS

- *Congestion price update*: The congestion prices are updated as follows:

$$q_l(t+1) = \left(q_l(t) + \alpha \Delta q_l(t)\right)^+, \tag{5}$$

where

$$\Delta q_l(t) = \sum_{j \in I(l)} \left[\sum_{k=1}^{K} H_k^j \int_t^{t+1} \frac{n_k(t) x_k(t)}{c_j} dt - 1_{j \in \mathcal{S}(t)} \right],$$

and $\mathcal{S}(t)$ denotes the set of links scheduled to transmit during the slot t.

- *User rate update:* The data rate of type-k users are

updated as follows:

$$x_k(t+1) = \min\left\{ \frac{1}{\sum_{l\in E} q_l(t+1) \sum_{j\in I(l)} \frac{H_k^j}{c_j}}, M_k \right\}, \quad (6)$$

where M_k is the maximum data rate of type-k users.

- *Transmission scheduling:* The link transmissions are scheduled in accordance with the *maximal scheduling policy*, i.e., subset of links M chosen for transmission during any slot satisfies that for each link $l \in E$, either $I(l) \cap M \neq \Phi$ or $q_l \leq 1$. For the sake of concreteness, we will assume that for link l to be scheduled during slot t, its congestion price $q_l(t)$ must be greater than 1.

CCS is similar in spirit to the joint congestion control and scheduling algorithm proposed in [10] under the node-exclusive interference model. However, our algorithm is significantly more general and works for all contention matrix based interference models, including the node-exclusive interference model. Some salient features of our algorithm are worth noting: (i) the congestion price of a link depends not only on its own backlog, but also on that of the links belonging to its interference set; (ii) the data rate of type-k users depends on the congestion prices of all those links that either belong to the route of type-k users or interfere with such a link. It is this proper setting of the user rates and congestion prices that allows CCS to achieve a fraction $d_I(G)$ of the capacity region, as stated in the following theorem:

Theorem 2: If the stepsize α is chosen to be small enough, CCS stabilizes the network for all user arrival rate vectors that belong to $\Omega^o/d_I(G)$, where Ω^o denotes the interior of set Ω.

Proof: The main difficulty in the proof is to construct an appropriate Lyapunov function that exhibits the required negative drift provided the user arrival rate vector belongs to $\Omega^o/d_I(G)$. We shall use the Lyapunov function $V(\vec{n},\vec{q}) := V_n(\vec{n}) + V_q(\vec{q})$, where $V_n(\vec{n}) := \sum_{k=1}^K \frac{\beta n_k^2}{2\lambda_k}$ and $V_q(\vec{q}) := \sum_{l\in E} \frac{q_l^2}{2\alpha}$. Let

$$\Delta V_q := E\left[V_q(\vec{q}(t+1)) - V_q(\vec{q}(t))|\vec{n}(t),\vec{q}(t)\right]$$

and

$$\Delta V_n := E\left[V_n(\vec{n}(t+1)) - V_n(\vec{n}(t))|\vec{n}(t),\vec{q}(t)\right].$$

Since all scheduled links l must have a congestion price $q_l > 1$, the projection operator in (5) is not required provided $\alpha \leq 1$. Now,

$$\begin{aligned}\Delta V_q &= \sum_{l\in E} \frac{1}{2\alpha} E\left[q_l^2(t+1) - q_l^2(t)|\vec{n}(t),\vec{q}(t)\right] \\ &= \sum_{l\in E} E\left[q_l(t)\Delta q_l(t) + \frac{\alpha}{2}(\Delta q_l(t))^2|\vec{n}(t),\vec{q}(t)\right] \\ &\leq \sum_{l\in E} q_l(t) \sum_{j\in I(l)} \sum_{k=1}^K \frac{H_k^j}{c_j} \int_t^{t+1} E\left[n_k(t)x_k(t)|\vec{n}(t),\vec{q}(t)\right] dt \\ &\quad - \sum_{l\in E} q_l(t) 1_{\{q_l(t)>1\}} + \frac{\alpha}{2} E\left[(\Delta q_l(t))^2|\vec{n}(t),\vec{q}(t)\right]. \end{aligned} \quad (7)$$

Using some algebraic manipulations, it can be shown that

$$\begin{aligned} E\left[(\Delta q_l(t))^2|\vec{n}(t),\vec{q}(t)\right] &\leq |E| \\ &\quad + C_L C_K \int_t^{t+1} E\left[n_k^2(t)x_k^2(t)|\vec{n}(t),\vec{q}(t)\right] dt, \end{aligned} \quad (8)$$

where

$$C_L = \max_{l\in E} \sum_{k=1}^K \sum_{j\in I(l)} \frac{H_k^j}{c_j} \text{ and } C_K = \max_{k=1,2,\ldots,K} \sum_{l\in E} \sum_{j\in I(l)} \frac{H_k^j}{c_j}.$$

Combining (7) and (8), we get

$$\begin{aligned}\Delta V_q &\leq \frac{\alpha C_L C_K}{2} \sum_{k=1}^K \int_t^{t+1} E\left[n_k^2(t)x_k^2(t)|\vec{n}(t),\vec{q}(t)\right] dt \\ &\quad + \frac{\alpha}{2}|E| - \sum_{l\in E} q_l(t) 1_{\{q_l(t)>1\}} + \\ &\quad \sum_{l\in E} q_l(t) \sum_{j\in I(l)} \sum_{k=1}^K \frac{H_k^j}{c_j} \int_t^{t+1} E\left[n_k(t)x_k(t)|\vec{n}(t),\vec{q}(t)\right] dt. \end{aligned} \quad (9)$$

Following the line of analysis in [10] used to prove Theorem 7, it can be shown that

$$\begin{aligned}\Delta V_n &\leq \sum_{l\in E} q_l(t) \sum_{j\in I(l)} \sum_{k=1}^K \frac{\beta\lambda_k H_k^j}{\mu_k c_j} \\ &\quad - \sum_{l\in E} q_l(t) \sum_{j\in I(l)} \sum_{k=1}^K \frac{H_k^j}{c_j} \int_t^{t+1} E\left[n_k(t)x_k(t)|\vec{n}(t),\vec{q}(t)\right] dt \\ &\quad - (\beta-1) \sum_{k=1}^K \int_t^{t+1} E\left[n_k(t)|\vec{n}(t),\vec{q}(t)\right] dt + C_1 \\ &\quad - \sum_{k=1}^K \frac{\mu_k}{4\lambda_k M_k} \int_t^{t+1} E\left[n_k^2(t)x_k^2(t)|\vec{n}(t),\vec{q}(t)\right] dt \end{aligned} \quad (10)$$

Observe that

$$\sum_{l\in E} q_l(t) 1_{\{q_l(t)>1\}} \geq \sum_{l\in E} q_l(t) - |E|. \quad (11)$$

Also, for all $\lambda \in \Omega^o/d_I(G)$, we have

$$\sum_{j\in I(l)} \sum_{k=1}^K \frac{\lambda_k H_k^j}{\mu_k c_j} < 1 \text{ for all } l \in E. \quad (12)$$

Using (9)-(12), it follows that given any $\epsilon > 0$ we can choose $\beta > 1$ and $\alpha > 0$ such that

$$\Delta V_n + \Delta V_q \leq C_2 - \epsilon \left(\sum_{l\in E} q_l(t) + \int_t^{t+1} E\left[n_k(t)|\vec{n}(t),\vec{q}(t)\right] dt \right),$$

where $C_2 = C_1 + \left(1 + \frac{\alpha}{2}\right)|E|$. The result follows using Theorem 2 in [17] and observing that $Q_l(t) = q_l(t)/\alpha$ for $l \in E$. ■

Theorem 1 lower bounds the performance of CCS in arbitrary wireless networks. We now consider wireless networks whose connectivity graph is a geometric graph and analyze the performance of CCS in such networks. The motivation for this is that if all transmissions in the network employ the same power level and the statistical properties of the noise are same at each node, then the connectivity graph of the

network is indeed a geometric graph. We shall further restrict our attention to K-hop interference models, so as to enforce some structure on the interference constraints.

Proposition 1: Consider a wireless network whose underlying connectivity graph is a geometric graph. If the interference constraints correspond to a K-hop interference model for some $K \geq 1$ and the stepsize α is chosen to be small enough, then CCS stabilizes the network for all user arrival rate vectors belonging to $\Omega^o/2$ for $K = 1$ and $\Omega^o \lfloor K/2 \rfloor^2 / (2K+1)^2$ for $K \geq 2$.

Proof: It is shown in [22] that if the underlying connectivity graph is a geometric graph and interference constraints correspond to a K-hop interference model, then $d_I(G)$ satisfies:

$$d_I(G) \leq \begin{cases} 2 & \text{for } K = 1 \\ \dfrac{(2K+1)^2}{\lfloor K/2 \rfloor^2} & \text{for } K \geq 2 \end{cases}$$

Substituting the above in Theorem 1 yields the desired result. ∎

Similar constant factor performance bounds can be shown to hold for disk graphs and (r, s)-civilized graphs.

B. Implementation Issues and Extensions

In this section, we identify some implementation issues concerning CCS and discuss how some of them can be dealt with.

The congestion price update in (5) requires each link to track the changes in the backlogs at all its interfering links. This can be done with the help of local message exchanges, where each link floods the relevant information (e.g., whether or not it was scheduled to transmit in the previous slot) within a local neighborhood that includes all its interfering links. Under an arbitrary contention matrix based interference model, in which a pair of links can interfere with each other even if they are several hops apart, the congestion price update would incur a considerable amount of overhead. In practice, however, the scope of interference is limited to a few hops (usually, 1 or 2). In such settings, the overhead due to congestion price update is expected to be small. A detailed discussion of this issue in the context of specific interference models can be found in Section V.

Further, the congestion price update assumes that a user rate update is instantaneously applied at all links on its route. In practice, however, there is some amount of time required to signal such a change to all links on the route. Moreover, in practice, user clocks might not be perfectly synchronized with each other. Thus, the user rate updates might be asynchronous. Both these issues can be dealt with following the line of analysis in [3]. In [3], the authors consider such issues in the context of a joint congestion control and scheduling algorithm under the node-exclusive interference model.

The rate update for type-k users requires the knowledge of congestion prices at all those links that either belong to their route or interfere with such a link. Note that this kind of behavior is common to all end-to-end congestion control protocols, like, for example, TCP and can be seen even under a wireline setting. In a wireline setting, however, there is no interference between a pair of links scheduled at the same time, and therefore the rate assigned to a flow is only a function of the congestion prices at the links on its route. In view of the results in [12], [3], one would expect that the stability properties of CCS should be preserved even when the user rate updates are performed at a much slower time scale as compared to the congestion price updates and also to a asynchronous setting. This issue will be addressed in our future work.

An important issue that remains to be addressed is that of scheduling the link transmissions. How can one generate a schedule with the properties listed in Section IV-A in a distributed fashion? Generating such a schedule might be computationally too expensive under general contention matrix based interference models. In the next section, we provide two randomized distributed algorithms that compute such a schedule in $\Theta(\Delta \log^2 |V|)$ rounds of computation and local message exchange under 1-hop and 2-hop interference models, where Δ denotes the maximum node degree in the network.

V. Randomized Distributed Algorithms for Maximal Scheduling Policy

Although, the maximal scheduling policy has been considered in many of the previous works [10], [4], [22], the difficulties that arise in implementing it in a distributed fashion in the presence of interference have not previously been dealt with. We now provide two randomized distributed algorithms for implementing the maximal scheduling policy under 1-hop and 2-hop interference models. Both these algorithms are inspired by a classical algorithm for constructing an maximal independent set in [18].

We start with describing our distributed computing model. As in [6] and other related works, we assume a *synchronous message passing distributed computing model*, which is a variation of the standard models used in the distributed computing literature. The main point of difference is the broadcast nature of the model which is typical of wireless networks. More precisely, we model the distributed computing architecture as a graph with undirected edges (we assume bidirectional links between the nodes as specified in the 1-hop and 2-hop interference models). Each node has a unique ID. The clocks at all the nodes are synchronized and the communication takes place in rounds, each occupying a slot. A packet transmission from node u is heard by all nodes v in its neighborhood, unless the node v itself transmits or some other neighbor of node v also transmits. It is this interference between simultaneous transmissions that makes it difficult to implement a maximal scheduling policy in wireless networks.

We now define some terminology that will be used in the sequel:

- uv: An undirected link between nodes u and v.
- $N(u)$: The set of neighbors of node u, i.e., nodes $v \in V$ such that $uv \in E$.
- $N^Q(u)$: The set of nodes $v \in N(u)$ such that $q_{(u,v)} > 1$.
- $d(u)$: $\max_{v \in N(u)} |N(v)|$.

- $d^Q(u)$: $\max_{v \in N^Q(u)} |N^Q(v)|$.
- $N_2(u)$: The set of two hop neighbors of node u, i.e., $\cup_{v \in N(u)} N(v)$.
- $N_2^Q(u)$: The set of undirected links $vw \in E$ such that $v \in N(u)$ and $\max(q_{(u,w)}, q_{(w,u)}) > 1$.
- $d_2^Q(u)$: $\max_{v \in N_2(u)} |N_2^Q(v)|$.

A. Maximal Scheduling under 1-Hop Interference Model

In this section, we propose a randomized distributed algorithm, namely MaxScheduleOneHop, that implements the maximal scheduling policy under the 1-hop interference model. MaxScheduleOneHop uses three subroutines, namely UpdatePrices, CompAndDistNeighborhoods, and UpdateAndDistNeighborhoods. As the names suggest, these schedules are used for the following purpose:

1. **UpdatePrices:** As required by the CCS, the congestion prices of the links should be updated at every slot. This subroutine allows each node in the network to update the congestion prices of its outgoing links.
2. **CompAndDistNeighborhoods:** This subroutine allows each node v to calculate $N^Q(v)$ and $d^Q(v)$ at every slot based on current congestion prices.
3. **UpdateAndDistNeighborhoods:** This subroutine allows each node to remove those nodes from its current neighborhood that were scheduled to transmit or receive during the current phase of MaxScheduleOneHop. More precisely, each node computes $N^Q(v)$ and $d^Q(v)$ considering only those nodes that have not been scheduled to transmit or receive as yet.

We are now ready to describe MaxSceduleOneHop:

MaxScheduleOneHop$(G, q(t))$

1. q(t+1) := **UpdatePrices(**$G, q(t)$**)**;
2. **CompAndDistNeighborhoods(**$G, q(t+1)$**)**;
3. $\mathcal{S}_0(t+1) := \phi$ and $b(u) = -1$ for all $u \in V$.
4. **for** $p = 1$ to $\lceil C_P \log |V| \rceil$ **do**
5. $\quad \mathcal{S}_p(t+1) := \mathcal{S}_{p-1}(t+1)$;
6. $\quad$ **for** $i = 1$ to $C_I \log |V|$ **do**
7. $\quad\quad$ Each node u with $b(u) = -1$ chooses to transmit with probability $\frac{1}{d^Q(u)+1}$. Upon deciding to transmit, it chooses a node v at random from $N^Q(u)$ and sends a RTS message to node v.
8. $\quad\quad$ If the RTS packet is successfully received by node v, it responds with a CTS message and sets $b(v) = 1$.
9. $\quad\quad$ Upon receiving the CTS packet, node u sets $b(u) = 1$. $\mathcal{S}_p(t+1) := \mathcal{S}_p(t+1) \cup uv$.
10. $\quad$ **end for**
11. **UpdateAndDistNeighborhoods(**$G, \mathcal{S}_{p-1}(t+1), \mathcal{S}_p(t+1)$**)**;
12. **end for**

The subroutines used by MaxScheduleOneHop will be described next, beginning with UpdatePrices,.

UpdatePrices(G, q)

1. **for each** $v \in V$ **do**
2. $\quad$ **if there exists** $uv \in \mathcal{S}(t)$
3. $\quad\quad$ **ReliablyBroadcast**$(v, \text{matched to } u)$;
4. $\quad$ **end it there exists**
5. $\quad$ Compute the new congestion prices of your outgoing links based on the local knowledge of $\mathcal{S}(t)$.
6. **end for each**

UpdatePrices allows each node to compute the congestion prices of its outgoing links based on the local knowledge of the schedule $\mathcal{S}(t)$. More precisely, each node that was involved in a reception or transmission during slot t, broadcasts this information to its neighbors using subroutine ReliablyBroadcast, which will be explained in a short while. Using this information and the local knowledge of user routes and data rates each node computes the current congestion price for all its outgoing links. Indeed, under the 1-hop interference model, we have

$$\Delta q_{(u,v)}(t) = \sum_{w \in N(u)} \Delta Q_{(u,w)} + \sum_{w \in N(v) \setminus u} \Delta Q_{(v,w)},$$

where

$$\Delta Q_{(u,w)} = \sum_{k=1}^{K} H_k^{(u,w)} \left[\int_t^{t+1} \frac{n_k(t) x_k(t)}{c_{(u,w)}} dt - 1_{(u,w) \in \mathcal{S}(t)} \right] + \sum_{k=1}^{K} H_k^{(w,u)} \left[\int_t^{t+1} \frac{n_k(t) x_k(t)}{c_{(w,u)}} dt - 1_{(w,u) \in \mathcal{S}(t)} \right].$$

Node u can compute $\Delta q_{(u,v)}(t)$ for each of its outgoing links (u, v), provided it has the local knowledge of user routes and data rates (we assume that this knowledge is maintained by the congestion control component) and knows which of its neighboring nodes were scheduled during slot t. The later information is provided to it by each of its neighboring node v using the subroutine ReliablyBroadcast that is described below.

ReliablyBroadcast$(v, data)$

1. **for** $k = 1$ to $\lceil C_K \Delta \log |V| \rceil$ **do**
2. $\quad$ Broadcast **data** with probability $\frac{1}{d(v)+1}$ to your neighbors.
3. **end for**

Next, we describe the subroutine CompAndDistNeighborhoods.

CompAndDistNeighborhoods(G, q)

1. **for each** $v \in V$ **do**
2. $\quad$ Compute $N^Q(v)$.
3. $\quad$ **ReliablyBroadcast**$(v, |N^Q(v)|)$;
4. $\quad$ Compute $d^Q(v)$.
5. **end for each**

CompAndDistNeighborhoods allows each node to compute $N^Q(v)$ and $d^Q(v)$. The computation of $N^Q(v)$ is straightforward. Once this is done, each node v uses the subroutine ReliablyBroadcast to broadcast $N^Q(v)$ to its neighbors; and based on this knowledge each node v computes $d^Q(v)$. Next, we describe the subroutine UpdateAndDistNeighborhoods.

UpdateAndDistNeighborhoods(G, S_{pr}, S_{cr})

1. **for each** $v \in V$ **do**
2. **if there exists** $uv \in S_{cr} \backslash S_{pr}$
3. **ReliablyBroadcast**$(v, \text{matched to } u)$;
4. **end if there exists**
5. Compute $N^Q(v)$ considering only those nodes that are currently unmatched.
6. **ReliablyBroadcast**$(v, |N^Q(v)|)$;
7. Compute $d^Q(v)$.
8. **end for each**

UpdateAndDistNeighborhoods allows each node to update $N^Q(v)$ and $d^Q(v)$ at the end of each phase in MaxScheduleOneHop. If a node was matched during the current phase, it broadcasts this information to its neighbors using the subroutine ReliablyBroadcast. With this knowledge, each node v updates $N^Q(v)$ by deleting those neighboring nodes which were matched during the current phase. Once this is done, each node v uses the subroutine ReliablyBroadcast to broadcast $|N^Q(v)|$ to its neighbors; and based on this knowledge each node v updates $d^Q(v)$.

We now show that if the constants C_P, C_I, and C_K are appropriately chosen the algorithm MaxScheduleOneHop returns a subset of links that satisfies the constraints imposed by the maximal scheduling policy with high probability. We start by analyzing the performance of ReliablyBroadcast:

Lemma 1: If every node uses the subroutine ReliablyBroadcast and $C_K \geq 8e$, then with probability at least $1 - \frac{1}{|V|^2}$ each node will be able to send its "data" to all its neighbors.

Proof: Consider a pair of neighboring nodes u and v in the network. During iteration k, node u broadcasts its data with a probability of $1/(d(u)+1)$. Node v will successfully receive this data if it decides not to transmit during iteration k and no node belonging to $N(v)\backslash u$ transmits during iteration k. Since each node $w \in N(v) \cup v$ satisfies $d(w) \geq |N(v)|$, each of these nodes transmits with probability no greater than $1/(|N(v)|+1)$ during iteration k. Thus, node v successfully receives the data from node u during iteration k with a probability $\mathbb{P}_k(u,v)$ that satisfies

$$\begin{aligned}\mathbb{P}_k(u,v) &\geq \frac{1}{|d(u)|+1}\left(1-\frac{1}{|N(v)|+1}\right)^{|N(v)|} \\ &\geq \frac{e^{-1}}{d(u)+1}.\end{aligned}$$

Noting that $\Delta = \max_{u \in V} d(u)$, we have that the probability $\mathbb{P}(u,v)$ that node v does not successfully receive the data from node u during any of the $\lceil C_K \Delta \log |V| \rceil$ iterations satisfies

$$\begin{aligned}\mathbb{P}(u,v) &\leq \left(1-\frac{e^{-1}}{d(u)+1}\right)^{C_K \Delta \log |V|} \\ &\leq e^{-\frac{e^{-1} C_K \Delta \log |V|}{\Delta+1}} \leq \frac{1}{|V|^4},\end{aligned}$$

provided $\Delta \geq 1$ and $C_K \geq 8e$. Now, summing over all possible node pairs[‡] and using the union bound, we obtain the desired result. ■

We are now ready to analyze the performance of MaxScheduleOneHop:

Proposition 2: If $C_P > 1$, $C_I \geq 6e^2$, and $C_K \geq 8e$, then with probability no smaller than $1 - \frac{2}{|V|}$ the set of edges returned by MaxScheduleOneHop satisfies the constraints imposed by the maximal scheduling policy.

Proof: Using Lemma 1, it follows that if $C_K \geq 8$ the prices and neighborhoods computed using UpdatePrices and CompAndDistNeighborhoods will be correct with probability at least $1 - \frac{2}{|V|^2}$. Observe that UpdateAndDistNeighborhoods is used $C_P \log |V|$ times by MaxScheduleOneHop and each time it uses ReliablyBraodcast twice to update and distribute the neighborhoods. Thus, with probability at least $1 - \frac{2C_P \log |V|}{|V|^2} \geq 1 - \frac{1}{|V|}$ for large enough $|V|$, the neighborhoods computed using UpdateAndDistNeighborhoods will be correct each time it is used. Next, we restrict our attention to those cases in which all of the above subroutines work correctly.

Consider the execution of MaxScheduleOneHop during iteration i of phase p under such a case. Let Δ_p denote the maximum node degree in the subgraph G_p of G obtained by removing all nodes in G that have already been matched at the beginning of phase p along with their outgoing and incoming links. Now, consider a node u with $|N^Q(u)| \geq \Delta_p/2$, where $N^Q(u)$ is the neighborhood of node u in G_p. The link (u,v) will be added to $\mathcal{S}_p$ during iteration i provided node u decides to transmit during iteration i and all nodes in $N(u) \cup N(v) \backslash u$ decide not to transmit. Arguing on similar lines as in the proof of Lemma 1, the probability $P_i(u,v)$ is greater than

$$\begin{aligned}&\frac{1}{d^Q(u)+1}\left(1-\frac{1}{|N(u)|+1}\right)^{|N(u)|}\left(1-\frac{1}{|N(v)|+1}\right)^{|N(v)|-1} \\ &\qquad \geq \frac{e^{-2}}{d^Q(u)+1}.\end{aligned}$$

Since the events corresponding to node u being matched to node v or w with $v \neq w$ are disjoint, the probability $\mathbb{P}_i(u)$ that node u is matched during iteration i equals $\sum_{v \in N(u)} \mathbb{P}_i(u,v)$ and satisfies $\mathbb{P}_i(u) \geq \frac{N^Q(u) e^{-2}}{d^Q(u)+1} \geq e^{-2}/2$ since $|N^Q(u)| \geq \Delta_p/2$ and $\Delta_p = \max_{v \in V} d^Q(v)$. The probability that node u is not matched during phase p is therefore no greater than $\left(1-\frac{e^{-2}}{2}\right)^{C_I \log |V|} \leq \frac{1}{|V|^3}$ provided $C_I \geq 6e^2$. Using the union bound, it follows that all nodes v with $|N^Q(v)| \geq \Delta_p/2$ at the beginning of phase p will be matched during phase p with probability at least $1 - \frac{1}{|V|^2}$. Thus, with probability at least $1 - \frac{1}{|V|}$, the maximum node degree decreases by

[‡]There are less than $\leq |V|(|V|-1)$ distinct ordered pairs of nodes in the network.

a factor of 2 during each phase. Since $\Delta_1 \leq |V| - 1$, $\lceil \log |V| \rceil \leq C_P \log |V|$ phases, provided $C_P > 1$, are sufficient to ensure that the constraints imposed by the maximal scheduling policy are met. Once again, appealing to the union bound, it follows that with probability at least $1 - \frac{2}{|V|}$ the subset of edges returned by MaxScheduleOneHop satisfies the constraints imposed by the maximal scheduling policy. ■

Each execution of UpdateAndDistNeighborhoods involves $\Theta(\Delta \log |V|)$ rounds of computation and local message exchange. Since MaxScheduleOneHop has $\Theta(\log |V|)$ phases and UpdateAndDistNeighborhoods is run at the end of each phase, MaxScheduleOneHop involves $\Theta(\Delta \log^2 |V|)$ rounds of computation and local message exchange.

Remark 1: If the maximum node degree in the network is significantly smaller than $|V| - 1$, then one can reduce the number of phases in MaxScheduleOneHop to $\lceil C_P \log \Delta \rceil$. MaxScheduleOneHop would then require $\Theta(\Delta \log \Delta \log |V|)$ rounds of computation and local message exchange. For example, if the maximum node degree in the network is $\Theta(\log |V|)$, as in the case of random geometric graphs, then by reducing the number of phases in MaxScheduleOneHop from $\Theta(\log |V|)$ to $\Theta(\log \log |V|)$, we can reduce its running time from $\Theta(\log^3 |V|)$ to $\Theta(\log^2 |V| \log \log |V|)$.

B. Maximal Scheduling under 2-Hop Interference Model

In this section, we propose a randomized distributed algorithm for implementing the maximal scheduling policy under the 2-hop interference model. Recall that under the 2-hop interference model no two links that are within two hops of each other can be scheduled to transmit or receive at the same time. The distributed computing model we adopt is the same as before. In particular, the model of interference between control packets is still the same. The reason for having a different interference model for data and control packets is that in most networks (e.g., IEEE 802.11 based networks) the control packets are usually much smaller in size than the data packets and are often transmitted at a much smaller rate than the data packets. Correspondingly, successful reception of a control packet requires much less SINR as compared to that of a data packet; thereby motivating the use of different interference models for each of them.

The algorithm we propose in this section, namely MaxScheduleTwoHop, is conceptually very similar to MaxScheduleOneHop. However, there are some additional difficulties that arise in case of 2-hop interference and are dealt with in MaxScheduleTwoHop. Indeed, a distinguishing feature of MaxScheduleTwoHop is the exchange of COL packets to ensure that no two links that are within two hops of each other decide to transmit or receive at the same time. More precisely, if a sender node detects an ongoing transmission while transmitting the RTS packet, it sends a subsequent *collision* (COL) packet. Successful reception of an RTS packet by the receiver guarantees that no other transmitter can be within one hop of the receiver. Further, no collision packet being sent guarantees that no two nodes that are within one hop of each other can decide to transmit at the same time. If the receiver does not hear a COL packet or a collision due to multiple such packets, it sends a *clear to send* (CTS) packet. If it detects an ongoing transmission while transmitting the CTS packet, it subsequently sends a COL packet. No collision packet being sent guarantees that no two nodes within one hop can decide to receive at the same time.

MaxScheduleTwoHop$(G, q(t))$

1. q(t+1) := **UpdateAndDistPrices(**$G, q(t)$**)**;
2. **CompAndDistTwoHopNeighborhoods(**$G, q(t+1)$**)**;
3. $\mathcal{S}_0(t+1) := \phi$ and $b(u) = -1$ for all $u \in V$.
4. **for** $p = 1$ to $C_P \log |V|$ **do**
5. $\quad \mathcal{S}_p(t+1) := \mathcal{S}_{p-1}(t+1)$;
6. $\quad$ **for** $i = 1$ to $C_I \log |V|$ **do**
7. $\quad\quad$ Each node u with $b(u) = -1$ chooses to transmit with probability $\frac{1}{d_2^Q(u)+1}$. Upon deciding to transmit, it chooses a node v at random from $N^Q(u)$ and sends a RTS message to node v.
8. $\quad\quad$ If a transmitting node detects any other transmission while transmitting, then it sends a COL packet immediately after the RTS packet.
9. $\quad\quad$ If a receiver node successfully receives the RTS packet and does not subsequently hear a COL packet (or a collision due to multiple such packets), then it sends a CTS message.
10. $\quad\quad$ If a receiver node v detects any other transmission while transmitting the CTS packet, then it sends a COL packet immediately after the CTS packet. Otherwise, it sets $b(v) = 1$.
11. $\quad\quad$ If a sender node u hears the CTS packet from its intended receiver but no subsequent COL packet, it sets $b(u) = 1$. $\mathcal{S}_p(t+1) := \mathcal{S}_p(t+1) \cup uv$.
12. $\quad$ **end for**
13. **UpdateAndDistTwoHopNeighborhoods(**$G, \mathcal{S}_{p-1}(t+1), \mathcal{S}_p(t+1)$**)**;
14. **end for**

MaxScheduleTwoHop uses three subroutines, namely UpdateAndDistPrices, CompAndDistTwoHopNeighborhoods, and UpdateAndDistTwoHopNeighborhoods that are very similar to their counterparts in case of MaxScheduleOneHop. The main difference, however, is that unlike their counterparts, these subroutines must broadcast information over two hops in order for each node v to be able to (i) compute the congestion price of its outgoing links, that requires the knowledge of local schedule over two hops; (ii) compute $N_2^Q(v)$ which requires the knowledge of congestion prices of links within two hops; and (iii) compute $d_2^Q(v)$ using the knowledge of $N_2^Q(w)$ for $w \in N_2(v)$. Broadcasting information over two hops is accomplished by using the subroutine ReliablyBroadcast twice in a series, once with your local information and then with the information obtained from your one hop neighbors. We are unable to provide detailed descriptions of these subroutines

due to want of space. These descriptions can be found in [20].

It can be shown (see [20] that MaxScheduleTwoHop requires $\Theta(\Delta \log^2 |V|)$ rounds of computation and local message exchange and with high probability returns a set of edges that satisfies the constraints imposed by the maximal scheduling policy under the 2-hop interference model.

Remark 2: If the maximum node degree in the network is significantly smaller than $|V| - 1$, then the number of phases in MaxScheduleTwoHop can be reduced to $\lceil C_P \log \Delta \rceil$. MaxScheduleTwoHop then requires $\Theta(\Delta \log |V| \log \Delta)$ rounds of computation and local message exchange.

VI. CONCLUDING REMARKS

In this paper, we considered the problem of throughput-optimal cross-layer design of wireless networks. We provided an upper bound on the capacity region of wireless networks under all contention matrix based interference models and proposed a joint congestion control and scheduling algorithm that stabilizes the network for all user arrival rate vectors that are within a fraction $1/d_I(G)$ of the capacity region. We have previously shown that $d_I(G)$ can be upper bounded by 49 in case of geometric graphs, provided the interference constraints are modeled using some K-hop interference model.

We also proposed two randomized distributed algorithms for implementing the maximal scheduling policy under 1-hop and 2-hop interference models. Both these algorithms require $\Theta(\Delta \log^2 |V|)$ rounds of computation and local message exchange. Although, many of the previous works have studied the performance of maximal scheduling policy under various network settings, it is for the first time in the literature that the difficulties involved in implementing the maximal scheduling policy in a distributed fashion in the presence of interference have been discussed in detail and fully accounted for in the analysis.

In light of our results, it appears that a distributed implementation of maximal scheduling would incur a significant overhead in networks where the maximum node degree is large. Thus, in such networks, the throughput provided by the maximal scheduling policy might be significantly smaller than the one calculated by ignoring the implementation overhead.

It is intriguing to ask the question: Is there a scheduling policy that is amenable to distributed implementation and can provide better throughput guarantees than the maximal scheduling policy under various interference models? We plan to address such issues in our future work.

REFERENCES

[1] D. J. Baker, J. Wieselthier, and A. Ephremides. A Distributed Algorithm for Scheduling the Activation of Links in a Self-organizing Mobile Radio Network. In *IEEE ICC*, pages 2F.6.1–2F.6.5, 1982.

[2] T. Bonald and L. Massoulie. Impact of fariness on Internet performance. In *ACM Sigmetrics*, 2001.

[3] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu. Joint Asynchronous Congestion Control and Distributed Scheduling for Multi-Hop Wireless Networks. In *IEEE INFOCOM*, 2006.

[4] P. Chaporkar, K. Kar, and S. Sarkar. Throughput Guarantees Through Maximal Scheduling in Wireless Networks. 43rd Annual Allerton Conf. on Communications, Control, and Computing, Sep 2005.

[5] R. L. Cruz and A. V. Santhanam. Optimal routing, link scheduling and power control in multihop wireless networks. In *INFOCOM*, 2003.

[6] H. Balakrishnan et al. The Distance-2 Matching Problem and its Relationship to the MAC-layer Capacity of Ad Hoc Wireless Networks. *IEEE JSAC*, 22(6):1069–1079, Aug 2004.

[7] B. Hajek and G. Sasaki. Link Scheduling in Polynomial Time. *IEEE Transactions on Information Theory*, 34(5):910–917, Sep 1988.

[8] F. Kelly, A. Maulloo, and D. Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. In *Journal of the Oper. Res. Society*, volume 49, pages 237–252, 1998.

[9] X. Lin and N. B. Shroff. Joint Rate Control and Scheduling in Multi-hop Wireless Networks. In *IEEE CDC*, 2004.

[10] X. Lin and N. B. Shroff. The Impact of Imperfect Scheduling on Cross-Layer Rate Control in Multihop Wireless Networks. In *IEEE INFOCOM*, Mar 2005.

[11] X. Lin, N. B. Shroff, and R. Srikant. On the Connection-Level Stability of Congestion-Controlled Communication Networks. Preprint, 2006.

[12] S. H. Low and D. E. Lapsley. Optimization flow control — I: basic algorithm and convergence. *IEEE/ACM Trans. on Networking*, 7(6):861–874, 1999.

[13] S. H. Low and R. Srikant. A Mathematical Framework for Designing a Low-Loss, Low-Delay Internet. *Network and Spatial Economics*, 4(1):75–102, March 2004.

[14] B. Miller and C. Bisdikian. *Bluetooth Revealed: The Insider's Guide to an Open Specification for Global Wireless Communications*. Prentice Hall, 2000.

[15] M. J. Neely, E. Modiano, and C. Li. Fairness and optimal stochastic control for heterogeneous networks. In *IEEE INFOCOM*, 2005.

[16] M. J. Neely, E. Modiano, and C. E. Rohrs. Dynamic Power Allocation and Routing for Time Varying Wireless Networks. In *IEEE INFOCOM*, 2003.

[17] M. J. Neely, E. Modiano, and C. E. Rohrs. Power allocation and routing in multibeam satellites with time-varying channels. *IEEE/ACM Trans. Netw.*, 11(1):138–152, 2003.

[18] D. Peleg. *Distributed Computing: A Locality-Sensitive Approach*. SIAM Monographs on Discrete Math. Appl., SIAM, Philadelphia, 2000.

[19] S. Sarkar and L. Tassiulas. End-to-end bandwidth guarantees through fair local spectrum share in wireless ad-hoc networks. *IEEE Trans. on Automatic Control*, 50(9), Sep 2005.

[20] G. Sharma, R. R. Mazumdar, and N. B. Shroff. Joint Congestion Control and Distributed Scheduling for Throughput Guarantees in Wireless Networks. Technical Report, School of ECE, Purdue University, 2006.

[21] G. Sharma, R. R. Mazumdar, and N. B. Shroff. Maximum Weighted Matching with Interference Constraints. In *IEEE FAWN*, March 2006.

[22] G. Sharma, R. R. Mazumdar, and N. B. Shroff. On the Complexity of Scheduling in Wireless Networks. In *ACM MOBICOM*, 2006.

[23] R. Srikant. *The Mathematics of Internet Congestion Control*. Birkhauser, 2004.

[24] L. Stockmeyer and V. Vazirani. NP-completeness of some generalizations of the maximum matching problem. *Inform. Process. Lett.*, 15(1):14–19, 1982.

[25] A. L. Stolyar. Maximizing Queueing Network Utility subject to Stability: Greedy Primal-Dual Algorithm. *Queueing Systems*, 50(4):401–457, 2005.

[26] L. Tassiulas and A. Ephremides. Jointly optimal routing and scheduling in packet radio networks. *IEEE Trans. on Info. Theory*, 38(1):165–168, Jan 1992.

[27] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Trans. on Automatic Control*, 37(12):1936–1948, December 1992.

[28] X. Wu and R. Srikant. Bounds on the Capacity Region of Multi-hop Wireless Networks under Distributed Greedy Scheduling. In *IEEE INFOCOM*, 2006.

[29] L. Xiao, M. Johansson, and S. Boyd. Simultaneous routing and resource allocation via dual decomposition. *IEEE Trans. on Comm.*, 52(7):1136–1144, July 2004.

[30] H. Yaiche, R. Mazumdar, and C. Rosenberg. A game-theoretic framework for bandwidth allocation and pricing in broadband networks. *IEEE/ACM Trans. on Networking*, 8(5):667–678, Oct 2000.

[31] Y. Yi and S. Shakkottai. Hop-by-hop Congestion Control over a Wireless Multi-hop Network. In *IEEE INFOCOM*, March 2004.