Hindawi Publishing Corporation
EURASIP Journal on Wireless Communications and Networking
Volume 2007, Article ID 34716, 9 pages
doi:10.1155/2007/34716

Research Article

Distributed and Cooperative Link Scheduling for Large-Scale Multihop Wireless Networks

Kezhu Hong,[1] Yingbo Hua,[1] and Ananthram Swami[2]

[1] *Department of Electrical Engineering, University of California, Riverside, CA 92521, USA*
[2] *Army Research Laboratory, 2800 Power Mill Road, Adelphi, MD 20783, USA*

Received 9 May 2007; Revised 4 September 2007; Accepted 24 October 2007

Recommended by Z. Liu

A distributed and cooperative link-scheduling (DCLS) algorithm is introduced for large-scale multihop wireless networks. With this algorithm, each and every active link in the network cooperatively calibrates its environment and converges to a desired link schedule for data transmissions within a time frame of multiple slots. This schedule is such that the entire network is partitioned into a set of interleaved subnetworks, where each subnetwork consists of concurrent cochannel links that are properly separated from each other. The desired spacing in each subnetwork can be controlled by a tuning parameter and the number of time slots specified for each frame. Following the DCLS algorithm, a distributed and cooperative power control (DCPC) algorithm can be applied to each subnetwork to ensure a desired data rate for each link with minimum network transmission power. As shown consistently by simulations, the DCLS algorithm along with a DCPC algorithm yields significant power savings. The power savings also imply an increased feasible region of averaged link data rates for the entire network.

1. INTRODUCTION

In multihop wireless networks, there are three major functions that all directly affect the network throughput. They are routing, link scheduling, and power control. Routing is concerned with the distribution of data flows from sources to destinations. Link scheduling determines whether the link between any two nodes should be turned on or off during any given time interval. (We leave link data rate to be autonomously decided by each node in the multihop network. In practice, some constraint on the link data rates should be applied. Link scheduling will be discussed in more detail later.) Power control deals with the allocations of transmission power to all concurrent (cochannel) transmitters in any given time interval. Cross-layer optimization over routing, link scheduling, and power control to achieve the best possible network throughput may be ideal but is extremely complex even for a network of only (for example) ten nodes. One such example is available in [1], where a low-power approximation is needed even for a small network with a central processor. Other works on cross-layer optimization include [2–5]. Experience suggests that the complexity of an exact cross-layer optimization is too high for most real-time applications. A realistic approach is therefore to design routing, link scheduling, and power control individually while maintaining a cross layer perspective. Furthermore, for large-scale multihop networks, distributed algorithms for routing, link scheduling, and power control are necessary to ensure the scalability of computing complexity.

For routing, there exist centralized algorithms as well as distributed algorithms. These algorithms are well documented in [6] and the references therein. For example, if all nodes are aware of the network topology, a simple distributed routing algorithm is such that each departing packet from a node is forwarded to the nearest neighboring node towards its destination. This algorithm is fully scalable, as it is not affected by the network size or the pattern of the network traffic demand.

For power control, there are also centralized algorithms and distributed algorithms. A distributed power control algorithm for multihop networks was presented in [7]. This algorithm is based on the same idea previously proposed for cellular networks, [8, 9]. The distributed (optimal) power control algorithm is possible because of the *standard* interference function (i.e., linear cone type constraint) [9].

For link scheduling, there are two broad definitions in the literature. One definition is the allocation of data rate to each link in a given time slot. This definition of link scheduling is equivalent to power control when there is a one-to-one mapping between a set of effective linkwise data rates and a set of power allocations for all transmitters. This is true when each node has one omnidirectional antenna. The second definition of link scheduling is how a time/frequency band is shared among different links. Time-division multiple access (TDMA) and frequency-division multiple access (FDMA) are the two most common examples. In this paper, we follow the second definition of link scheduling. Combining this link scheduling and the conventional power control leads to what we call space-time power schedule [10] as shown later.

To our knowledge, there has been very little work on distributed link scheduling algorithms besides random access protocols such as ALOHA [11], carrier sense multiple access (CSMA) as in IEEE 802.11 [12], mesh mode distributed scheduling (MSH-DSCH) as in IEEE 802.16 [13], and their variations. These random access protocols do not provide a flexible control of the spacing between concurrent cochannel transmissions for multihop networks. CSMA in principle prevents all concurrent cochannel transmissions within an entire radio transmission range centered at each receiver, which generally causes the number of concurrent cochannel transmissions to be very sparse in a large network. The MSH-DSCH in principle allows concurrent cochannel transmissions that are two hops away from each other, but its optimality has not been established. The spacing between concurrent cochannel transmissions is known to be crucial for maximal throughput of large multihop networks, [7, 14, 15].

The link scheduling algorithm proposed in [7] is centralized. The work [16] introduces a *locally centralized* scheme where the sharing of the time/frequency band of the entire network is predetermined and the scheduling is centrally optimized over each time/frequency band within which a subsystem operates. The spectral efficiency of this scheme diminishes as the network size increases and the power remains bounded. The work [17] presents a distributed scheduling scheme to resolve conflicts in multicasting. This scheme does not address the spacing issue of concurrent cochannel links in large networks. In fact, what can be achieved by that scheduling scheme can be treated as a very good initial condition for our link scheduling problem. The work [18] describes a distributed scheme to achieve interference-free scheduling. For large networks, interference-free scheduling is known to be highly inefficient [14]. The work [19] is based on graph coloring where only conflicting links are assigned orthogonal channels and the spacing issue is not addressed.

In this paper, we present a distributed and cooperative link scheduling (DCLS) algorithm, which works in the following context. The network is synchronous, in that, all data transmissions in the network are time framed. At the beginning of a time frame, each transmitting node looks for a nearby receiving node (or vice versa) through a random access protocol with short control packets. This leads to a set S of successful transceiver pairs (also called links) to be scheduled within the time frame. Then the set S undergoes a distributed and cooperative process dictated by the DCLS algorithm. At the end of this process, the set S is divided into several subsets of links, S_k, $k = 1, 2, \ldots, K$, where each subset consists of links properly spaced from each other. We will use *subset* and *subnetwork* interchangeably. For each subset, a distributed and cooperative power control (DCPC) algorithm such as in [7–9] is carried out. After the DCPC algorithm completes power control allocation for one subset, all links in the subset carry out data transmissions with the allocated power within a time slot in the time frame. The process repeats for each subset with a corresponding time slot. Both the DCLS and DCPC algorithms have fast convergence rates. Hence, the time required for link schedule and power control can be kept much smaller than the time required for data transmission. The entire time frame for link schedule, power control, and data transmission needs to be smaller than the channel coherence time. Such a condition should be first verified for each application in practice.

In Section 2, we present the DCLS algorithm in detail. In Section 3, we illustrate the performance of the DCLS algorithm in terms of the spacing between concurrent cochannel transmissions and the savings in network transmission power. The power saving not only is important for energy saving for low-energy nodes but also directly translates into an increased feasible region of averaged link data rates. The power saving may also imply a reduced dynamic power range of transceivers, which is particularly important to ensure that power amplifiers can be operated in the linear region.

2. DISTRIBUTED AND COOPERATIVE LINK SCHEDULING

Let us first consider the following formulation of what we call space-time power schedule for a given time frame:

$$\min_{\{P_l(k);\, l=1,2,\ldots,L;\, k=1,2,\ldots,K\}} \frac{1}{K}\sum_{k=1}^{K}\sum_{l=1}^{L} P_l(k)$$

$$\text{subject to } \frac{1}{K}\sum_{k=1}^{K}\log_2\left(1+\mathrm{SINR}_l(k)\right) \geq r_l, \qquad l = 1, 2, \ldots, L,$$

$$\mathrm{SINR}_l(k) = \frac{\left|g_{ll}\right|^2 P_l(k)}{\sum_{j=1;j\neq l}^{L}\left|g_{jl}\right|^2 P_j(k) + {\sigma_l}^2}, \tag{1}$$

where L is the total number of active links (transceiver pairs) in the time frame, K the number of time slots per frame, $P_l(k)$ the transmission power to be consumed by the transmitter of link l in time slot k, $\mathrm{SINR}_l(k)$ the signal-to-interference-and-noise ratio for link l and slot k, g_{ll} the channel gain of link l, g_{jl} the channel gain between the transmitter of link j and the receiver of link l, ${\sigma_l}^2$ the noise variance at the receiver of link l, and r_l the desired data rate in bits/second/Hertz of link l during the frame. The channel gains are assumed to be constant during the frame under consideration.

The problem (1) aims to minimize the total power consumption, while the required time-averaged data rate (a measure of quality of service) of each link is satisfied. Reducing power consumption is not only useful to preserve energy for low-energy nodes but also useful to meet practical constraints on the dynamic power range of transceivers. This is particularly true to ensure linearity of power amplification as required by, for example, orthogonal frequency division-multiplexing (OFDM) transceivers. The problem (1) is also a generalization of link scheduling and power control, the solution to which would be ideal. However, under $K > 1$ and $L > 1$, this problem is nonconvex and even a centralized algorithm can not guarantee the globally optimal solution. A centralized algorithm also becomes too complex as L increases while $K > 1$. It would be even harder, if not impossible, to develop a distributed algorithm to search for the global solution of (1).

We now step back to consider link scheduling and power control separately. To develop a distributed link scheduling algorithm, we now consider the following distributed optimization problem for each link l:

$$\begin{aligned} \min_{\{P_l(k);\,k=1,2,\dots,K\}} \quad & \frac{1}{K}\sum_{k=1}^{K} P_l(k) \\ \text{subject to} \quad & \frac{1}{K}\sum_{k=1}^{K}\log_2\left(1+\frac{\mathrm{SINR}_l(k)}{\lambda_l}\right) \ge r_l. \end{aligned} \tag{2}$$

The importance of the scalar λ_l will be explained later. For each fixed l, the problem of (2) is convex and has the following water-filling solution:

$$\widehat{P}_l(k) = \left(\nu_l - \lambda_l G_{ll}(k)\right)^+, \tag{3}$$

where ν_l is such that

$$\frac{1}{K}\sum_{k=1}^{K}\log_2\left(1+\frac{\widehat{P}_l(k)}{\lambda_l G_{ll}(k)}\right) = r_l, \tag{4}$$

$G_{ll}(k) \triangleq |g_{ll}|^{-2}P_{l,IN}(k)$, $P_{l,IN}(k) \triangleq \sum_{j=1;j\neq l}^{L}|g_{jl}|^2P_j(k)+\sigma_l^2$, and $(x)^+ = x, x > 0; 0, x \le 0$.

The above solution is based on the water-filling lemma: let r and $a_1, a_2, \dots, a_K$ be positive constants, and $x_1, x_2, \dots, x_K$ be nonnegative variables. The solution to $\min\sum_{k=1}^{K}x_k$ subject to $\sum_{k=1}^{K}\log(1+x_k/a_k) \ge r$ is given by $\widehat{x}_k = (\nu - a_k)^+$ for $k = 1,2,\dots,K$, where ν is such that $\sum_{k=1}^{K}\log(1+\widehat{x}_k/a_k) = r$.

In the solution (3), the transmitter[1] of link l must know $P_{l,IN}(k)$, the power of interference-plus-noise for link l, which depends on the power allocations for other links. To handle this problem, we can iterate the computation of (3) as follows. Before each iteration of (3), all links conduct interference calibration simultaneously, that is, the transmitter of link l uses the previous power allocations $P_l(k)$, $k = 1,2,\dots,K$, to transmit a short test signal over K short time slots, and the receiver of link l measures the values of $G_{ll}(k)$ for $k = 1,2,\dots,K$. More specifically, the signal received by the receiver of link l in the kth short time slot can be written as (in baseband form)

$$x_l(k,t) = g_{ll}\sqrt{P_l(k)}s_l(t) + \sum_{j=1;j\neq l}^{L} g_{jl}\sqrt{P_j(k)}s_j(t) + n_l(k,t). \tag{5}$$

for interference calibration, $s_j(t)$ for all j are short test signals. Each of the test signals is assumed to have unit power during its short duration. As discussed later, we can ensure that the test signal for link l is orthogonal to the test signals used for all links within a radius of dominant interference region. Then in (5), the signal component $g_{ll}\sqrt{P_l(k)}s_l(t)$ is orthogonal, at least approximately, to the interference term $\sum_{j=1;j\neq l}^{L} g_{jl}\sqrt{P_j(k)}s_j(t)$. Therefore, given $x_l(k,t)$ for $k = 1,2,\dots,K$ and $s_l(t)$, the receiver of link l can estimate g_{ll} by $(1/K)\sum_{k=1}^{K}E\{|s_l^*(t)x_l(k,t)|^2\}P_l(k)^{-1/2}$, where E denotes temporal averaging over t and the superscript $*$ denotes complex conjugation. The receiver of link l can then estimate $P_{l,IN}(k)$ by $E\{|x_l(k,t) - g_{ll}\sqrt{P_l(k)}s_l(t)|^2\}$. With $G_{ll}(k) \triangleq |g_{ll}|^{-2}P_{l,IN}(k)$, the receiver of link l can compute the new power allocations $\widehat{P}_l(k)$ for $k = 1,2,\dots,K$ using (3). Note that the link l needs to measure $G_{ll}(k)$ but not such individual components as $|g_{jl}|^2$, $P_j(k)$, or σ_l^2. Such a process of interference calibration is also needed for the distributed and cooperative power control (DCPC) algorithm [7–9].

Although the number L of active links within a time frame may grow with the network size, the number of dominant interferers to a given link does not grow, provided that the node density remains constant. This means that the number of orthogonal test signals for the entire network does not need to grow with the network size. As mentioned earlier, it is sufficient to ensure that the test signal transmitted by any node is orthogonal to the test signals transmitted by all nodes within a radius R_0 of this node. Denote by M the number of all orthogonal signals, and by N the maximum possible number of nodes within the radius R_0. Clearly, N can be much smaller than the total number of nodes in the network. We only need $M \ge N$. As long as the network topology is relatively stable, the test signals can be autonomously chosen by nodes as follows. Each node chooses a test signal at a random time. Once a test signal is chosen by a node, the identification of this test signal is immediately broadcasted to all nodes within the radius R_0. As long as the test signal chosen by a node is orthogonal to those already chosen by others within the radius R_0, a desired assignment of test signals will be achieved. Note that it is not necessary that the test signals by all nodes within the radius R_0 are orthogonal to each other.

During interference calibration, all exchanges of local information between the transmitter and the receiver of each link must be done locally via a random access protocol. Such local information includes the index of the test signal, channel state information, power allocation, and the desired date rate. As long as the amount of local information is much smaller than the amount of information in the data packets to be transmitted, the overhead caused by interference calibration can be tolerable.

[1] For convenience, we will often simply refer to link l for allocated transmission power.

However, although distributed, the algorithm of (3) does not necessarily converge to a meaningful result without a proper choice of λ_l. To study the convergence behavior of (3), let us rewrite (3) as

$$P_l^{(i)}(k) = \left(\nu_l^{(i)} - \frac{\lambda_l}{|g_{ll}|^2}\left(\sum_{j=1;j\neq l}^{L} |g_{jl}|^2 P_j^{(i-1)}(k) + \sigma_l^2\right)\right)^+, \tag{6}$$

where $P_l^{(i)}(k)$ is the power allocation, computed at iteration i, for link l and slot k, and $\nu_l^{(i)}$ needs to be computed to satisfy

$$\frac{1}{K}\sum_{k=1}^{K}\log_2\left(1 + \frac{|g_{ll}|^2 P_l^{(i)}(k)}{\lambda_l \sum_{j=1;j\neq l}^{L} |g_{jl}|^2 P_j^{(i-1)}(k) + \sigma_l^2}\right) = r_l. \tag{7}$$

Furthermore, we can write (6) as

$$P_l^{(i)}(k) = \left(\beta_l^{(i)} - \sum_{j=1;j\neq l}^{L} \alpha_{jl} P_j^{(i-1)}(k)\right)^+, \tag{8}$$

where $\alpha_{jl} = \lambda_l(|g_{jl}|^2/|g_{ll}|^2)$ and $\beta_l^{(i)} = \nu_l^{(i)} - \lambda_l(\sigma_l^2/|g_{ll}|^2)$.

The convergence of (8) still appears difficult to prove or disprove rigorously due to the interactive nature of power allocations among many links. But we can observe the followings from (8). The influence on the current power allocation (as function of k) at link l from the previous power allocation at link j is captured by the term $\alpha_{jl}P_j^{(i-1)}(k)$. This influence from link j is down weighted at each iteration if $\alpha_{jl} < 1$, but is up weighted at each iteration if $\alpha_{jl} > 1$.

We now assume that $\alpha_{jl} < 1$ for all $j \neq l$. Then the influence on the current power allocation, at each link from the previous power allocations at all other links, is down weighted after each iteration. This means that after each iteration, the power allocation at each link becomes closer to a value independent of k due to the first term in (8). Therefore, as the iteration continues (i.e., i becomes large), $P_l^{(i)}(k)$ becomes independent of k at each link l. This result is unfortunately not useful. Indeed, if all links consist of pairs of the nearest neighboring nodes, then we typically have $|g_{jl}|^2 < |g_{ll}|^2$ (i.e., the channel gain from the transmitter of link j to the receiver of link l is smaller than the channel gain from the transmitter of link l to the receiver of link l), and hence $\alpha_{jl} < 1$ for all $j \neq l$ if $\lambda_l = 1$. Therefore, in order to have a meaningful result from (8), we must have $\lambda_l > 1$.

We now consider a case where $\alpha_{lm} = \alpha_{ml} > 1$, $\alpha_{ml} > \alpha_{jl}$ for all $j \neq m$, and $\alpha_{lm} > \alpha_{jm}$ for all $j \neq l$. This case typically corresponds to a situation where the transmitter of link m is the closest to the receiver of link l, and vice verse. In this case, the influence on the power allocations at link l and link m at each iteration is dominated by the previous power allocations at these two links. We see from (8) that the current power allocation at link l is always complementary to the previous power allocation at link m (by ignoring the influence from all other links), and vice versa. For example, if the previous power allocation at link m is high in the first slot and low in the second slot, then the currently power allocation at link l becomes low in the first slot and high in the second slot. Since $\alpha_{lm} = \alpha_{ml} > 1$, the power allocation at each of the two links becomes more diverse (i.e., more fluctuating over k) as the iteration continues, provided that the initial power allocation at each link consists of distinct values over k. Because of (7), the averaged power at each link and each iteration should be bounded, provided that r_l, $l = 1, 2, \ldots, L$, are inside their feasible region. Therefore, as the iteration continues, eventually, the operator $(x)^+$ in (8) becomes effective and sets $P_l^{(i)}(k) = 0$ for some k. (Note that $P_l^{(i)}(k)$ cannot be zero for all k because of the condition (7).) This is a desired result for link scheduling because we want a set of nearby links to share the available spectrum orthogonally in order to have a high-spectral efficiency for the network.

However, once $P_l^{(i)}(k)$ becomes zero for some k at a value of i, will $P_l^{(i+1)}(k)$ bounce back from zero after another iteration? Our simulation confirms that the iteration of (8) does oscillate even after some components of $P_l^{(i)}(k)$ for $k = 1, 2, \ldots, K$ become zero, that is, they may bounce back and forth as the iteration of (8) continues. This is because of interactions between links. To solve this problem, we modify (6) as follows:

$$P_l^{(i)}(k) = \xi \cdot P_l^{(i-1)}(k) + (1-\xi) \cdot \widehat{P}_l^{(i)}(k), \tag{9}$$

where $\widehat{P}_l^{(i)}(k)$ is computed by (3) or equivalently (6), and $0 < \xi < 1$ is a memory factor. (The memory factor here is similar to what is often called a forgetting factor in the literature of adaptive signal processing.) With the memory factor, the above algorithm has a good convergence behavior as observed in simulation.

Furthermore, r_l in the above algorithm has lost its original meaning as a desired link data rate. In fact, it may be necessary to choose r_l to be different from the desired link data rate. Because the tuning parameter λ_l is typically larger than one, r_l often needs to be smaller than a desired link data rate to avoid possible nonconvergence of $P_l^{(i)}(k)$. Such a nonconvergence occurs when the values of r_l, $l = 1, 2, \ldots, L$, are outside their feasible region of (2). The feasible region decreases as λ_l, $l = 1, 2, \ldots, L$, increase. To understand why nonconvergence occurs when r_l, $l = 1, 2, \ldots, L$, are outside their feasible region, one only needs to see that no finite values of $P_l^{(i)}(k)$ can be a converged condition since it would otherwise suggest that r_l, $l = 1, 2, \ldots, L$, are feasible, that is, achievable by finite power allocations. In fact, such a nonconvergence should correspond to a divergence of some of the values of $P_l^{(i)}(k)$. However, a rigorous proof of such a divergence is complicated by the interactive nature of the iterations of (6) and (9).

As long as λ_l is large enough, upon convergence, $P_l^{(i)}(k)$ for each l becomes nonzero for some (at least one) of $k = 1, 2, \ldots, K$, and zero for the rest of $k = 1, 2, \ldots, K$. Then for each k, there is a subset S_k of the network, corresponding to all nonzero values of $P_l^{(i)}(k)$, that is, $S_k = \{1 \leq l \leq L \mid P_l^{(i)}(k) > 0\}$. For each S_k, one can apply the DCPC algorithm [7–9] to determine the actual power allocation for data transmission on each link $l \in S_k$ and slot k. The DCPC algorithm required here is essentially an algorithm to solve (1) with $K = 1$ and $l \in S_k$. This is a convex problem and the convergence is guaranteed. If link l is scheduled to be on for K_0 out of K slots, then the data rate for link l in one of the on-slots should be replaced by $(K/K_0)r_l$, where r_l is the desired data rate of link l averaged over all time slots.

To summarize, the distributed and cooperative link scheduling (DCLS) algorithm that we have developed is as follows.

Step 1. Once all links are formed for a time frame, each link l initializes randomly $P_l^{(0)}(k)$, $k = 1, 2, \ldots, K$, with K distinct nonzero values. Set $i = 1$.

Step 2. All links conduct interference calibration concurrently. The power transmitted by the transmitter of link l is given by $P_l^{(i-1)}(k)$, $k = 1, 2, \ldots, K$. Then at link l, the channel gain g_{ll} is estimated, and the received interference-plus-noise power $P_{l,IN}{}^{(i)}(k)$, $k = 1, 2, \ldots, K$, is measured. Set $G_{ll}{}^{(i)}(k) \triangleq |g_{ll}|^{-2} P_{l,IN}{}^{(i)}(k)$, $k = 1, 2, \ldots, K$.

Step 3. At each link l, update, for $k = 1, 2, \ldots, K$,

$$P_l^{(i)}(k) = \xi \cdot P_l^{(i-1)}(k) + (1 - \xi) \cdot \left(\nu_l^{(i)} - \lambda_l G_{ll}{}^{(i)}(k)\right)^+, \tag{10}$$

where $0 < \xi < 1$ and $\nu_l^{(i)}$ is such that $(1/K)\sum_{k=1}^{K} \log_2(1 + (P_l^{(i)}(k)/\lambda_l G_{ll}{}^{(i)}(k))) = r_l$.

Step 4. Until convergence, set $i = i + 1$ and go to Step 2. Upon convergence, the subsets of the networks are formed by $S_k = \{1 \le l \le L \mid P_l^{(i)}(k) > 0\}$, $k = 1, 2, \ldots, K$.

It is important to note that for any given $K > 1$, the value of λ_l influences the sparseness of the concurrent cochannel transmissions around link l. The desired sparseness may depend on the desired link rates. The choice of λ_l can be decided locally by each link l. In the next section, we will illustrate the impact of λ_l by letting $\lambda_l = \lambda$ for all l.

3. SIMULATIONS

In this section, we illustrate the performance of the DCLS algorithm. We consider several examples of network topology. Performance is measured by either the spacing between concurrent transmissions and/or the consumption of network transmission power.

The noise variance is $\sigma_l^2 = 1$ for all $l = 1, 2, \ldots, L$. The squared channel gain is $|g_{jl}|^2 = d_{jl}^{-3}$, where d_{jl} is the distance between the transmitter of link j and the receiver of link l. The total consumed power that we mention later is the total averaged power consumed by the transmitters of all links (averaged over all K slots).

3.1. Ring network

A ring network consisting of multiple links on a circle is shown in Figure 1. The distance between adjacent nodes is one. Also shown in Figure 1 is a typical partition of the ring network by the DCLS algorithm with $K = 2$ and a large enough λ. We see that the spacing between concurrent links in each time slot is ideal for the case where $K = 2$. If λ is not large enough, there is no partition of the network.

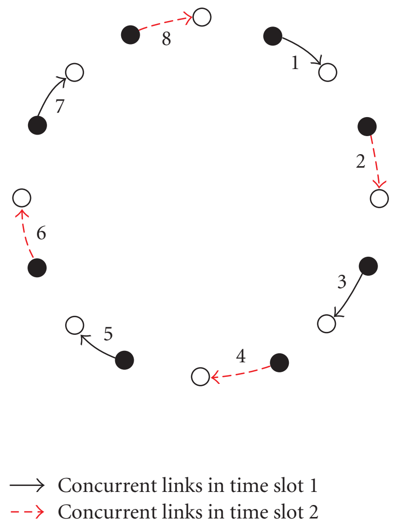
1
2
3
4
5
6
7
8
Concurrent links in time slot 1
Concurrent links in time slot 2

FIGURE 1: A ring network of 8 links is partitioned by the DCLS algorithm with $K = 2$ into two subnetworks of 4 links each. The (filled) black circles denote the transmitting nodes and the blank ones the receiving nodes.

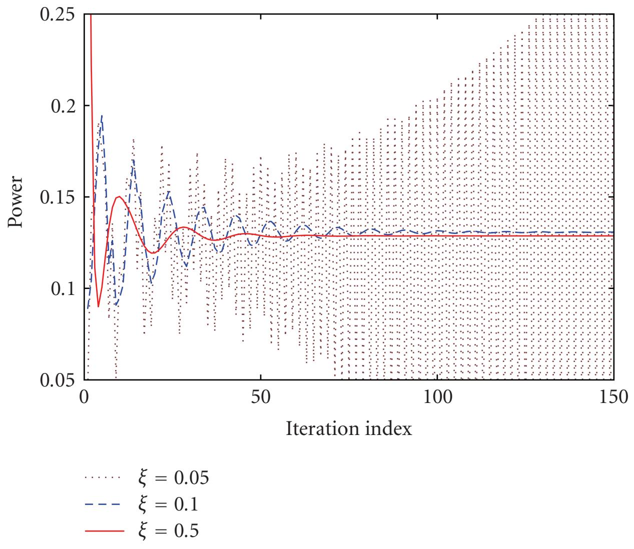

FIGURE 2: Illustration of $P_l^{(i)}(k)$ of the DCLS algorithm with $K = 3$ and $\lambda = 5$ for the ring network versus the iteration index i for an arbitrary choice of l and k and different values of ξ. This particular link l is scheduled to be *on* in the slot k since $P_l^{(i)}(k)$ converges to a nonzero value (for $\xi \ge 0.1$).

Figure 2 illustrates the importance of ξ in the DCLS algorithm to prevent oscillations. In all simulation cases, $\xi = 0.5$ is sufficient to ensure convergence. A mathematical proof of this convergence property remains open.

In Table 1, several outcomes (link schedules) of the DCLS algorithm are shown. Here, λ_s is such that if $\lambda > \lambda_s$, then at the convergence of the DCLS algorithm, each link is sched-

Table 1: Outcomes of the DCLS algorithm for the ring network of 8 links and different values of K.

No. of time slots	λ_s	Scheduled subsets
$K = 2$	2	$\{1,3,5,7\}$; $\{2,4,6,8\}$
$K = 3$	22	$\{1,4,7\}$; $\{2,6\}$; $\{3,5,8\}$
$K = 4$	32	$\{1,5\}$; $\{2,6\}$; $\{3,7\}$; $\{4,8\}$

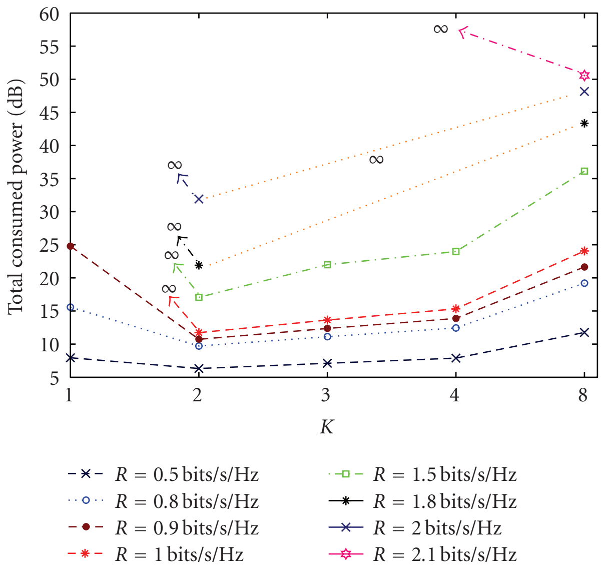

Figure 3: Total transmission power (dB) consumed by the ring network of eight links versus K, where ∞ indicates that there is no finite power allocation to satisfy the uniform link data rate $r_l = R$ in bits/s/Hz. Shown in this figure, $R \geq 1$ is infeasible for $K = 1$, $R \geq 1.8$ infeasible for $K = 3, 4$, and $R \geq 2.1$ infeasible for $K = 2$. The transition from a feasible data rate with moderate power allocations to an infeasible data rate with infinite power allocations is very sharp for each of $K < 8$.

uled to be on for only one slot and off for all other slots. We see that as K increases, so does λ_s. The value of λ_s is empirically established via simulation. The importance of λ_s for each given value of K is that if $\lambda > \lambda_s$, the maximal sparseness of concurrent cochannel transmissions is achieved under the given value of K. It should be obvious that the larger the value of K, the larger the maximum sparseness of concurrent cochannel transmissions.

Once the original set of links is partitioned into several subsets of links by the DCLS algorithm, each subset of links applies the DCPC algorithm [7–9] to schedule (optimal) transmission power to meet the desired data rates for all links in the subset. In Figure 3, we illustrate the total (transmission) power consumption by all links in all subsets versus K, assuming a uniform link date rate $r_l = R$. Both the data rate R and the transmission power shown in the figure are averaged over the whole time frame of K time slots.

We see from Figure 3 that for $K < 8$, the minimum power consumption is achieved at $K = 2$, or equivalently, the maximum feasible uniform data date is achieved at $K = 2$. We also see that for most of the feasible data rates under $K = 2$, the power consumption under $K = 2$ is smaller than that under $K = 8$. In other words, the transition from a feasible data rate with moderate power allocations to an infeasible data rate with infinite power allocations under $K = 2$ (in fact, under any $K < 8$) is very sharp. In Figure 3, $R = 2.0$ bits/s/Hz is highly feasible for $K = 2$, but $R = 2.1$ bits/s/Hz is infeasible for $K = 2$. In fact, at $R = 2.0$ bits/s/Hz, $K = 2$ still consumes less power than $K = 8$.

The sharp transition of feasible region is a common phenomenon due to interferences. Consider two concurrent cochannel (independent) links. The sum capacity of the two links is

$$C(P_1, P_2) = \log_2\left(1 + \frac{P_1}{1 + \alpha_2 P_2}\right) + \log_2\left(1 + \frac{P_2}{1 + \alpha_1 P_1}\right), \tag{11}$$

where the noise variance is normalized to one, and the channel gains are normalized to P_1, P_2, α_1, and α_2. The transmission powers of the two links are absorbed into P_1 and P_2. If the capacity of each link is lower bounded by a nonzero positive number, then both P_1/P_2 and P_2/P_1 are upper bounded, and hence the sum capacity is upper bounded by a constant regardless of how large P_1 and P_2 become. As the capacity of each link increases within its feasible region, both P_1 and P_2 must increase. But as P_1 and P_2 increase from moderate values to infinity, the change of $C(P_1, P_2)$ is small. For example, let $\alpha_1 = \alpha_2 = 0.5$. Then we have $(C(\infty,\infty) - C(10,10))/C(\infty,\infty) = \log_2(9/8)/\log_2(3) = 0.107$.

The case of $K = 8$ corresponds to the conventional TDMA scheduling which is interference free and has an infinite feasible region, provided that the power is unlimited. It is important to note that in order for $K = 8$ to be better than $K = 2$ for the ring network, we need a coding and modulation technique to achieve a peak (i.e., within a time slot) spectral efficiency larger than $2 \times 8 = 16$ bits/s/Hz for a single link because only one of the eight time slots is used for each of the eight links. Such a high-spectral efficiency is so far still a practical challenge at the physical layer design of transceivers (even for links with multiple antennas).

In IEEE 802.11, carrier sense multiple access (CSMA) is used to prevent concurrent cochannel transmissions within an entire radio transmission radius centered at each receiver. This radius depends on the transmission power from the transmitting nodes. For the ring network, if the transmission power is large enough, the effect of CSMA (after ignoring all overheads) would correspond to the choice $K = 8$ in our scheme. Clearly, when the uniform data rate is within most of the feasible region under $K = 2$, CSMA is much less efficient than our scheme. On the other hand, if the transmission power is not large enough for CSMA to prevent concurrent cochannel transmissions within the ring network, we effectively have a situation where $K < 8$. However, it is generally difficult to use CSMA to control the actual value of K and the required power from each transmitting node to ensure the desired data rates for all links.

3.2. *Square grid network*

Shown in Figure 4 is a square network of 72 links, involving 144 nodes evenly distributed on a square grid. The dis-

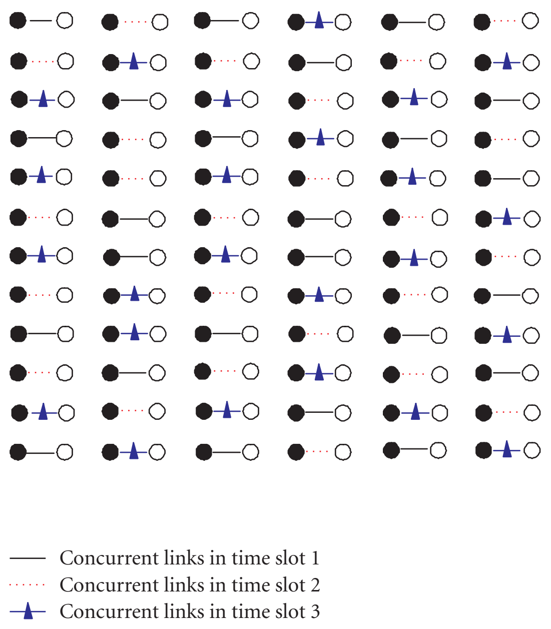
Concurrent links in time slot 1
Concurrent links in time slot 2
Concurrent links in time slot 3

FIGURE 4: A square network of 72 links with links partitioned, by the DCLS algorithm with $K = 3$, into three subsets corresponding to three time slots, where black circles represent transmitting nodes and blank circles denote receiving nodes.

tance between adjacent nodes is one. Also shown in Figure 4 is a typical outcome of the DCLS algorithm with $K = 3$. Although the exact outcomes from the DCLS algorithm may differ, depending on the initializations, the pattern of each subset has been found to be roughly the same. We see from the figure that although the outcome of the DCLS algorithm is not guaranteed to be optimal, most of the adjacent links are scheduled for transmission in different time slots, which is a desirable result for this network. Simulations have shown that for $K = 2, 3, 4$, the values of λ_s are 5, 18, 30, respectively.

Figure 5 illustrates $P_l^{(i)}(k)$ of the DCLS algorithm with $K = 4$ versus the iteration index i for $k = 1, 2, 3, 4$ and an arbitrary l. Here, convergence is achieved after 30 iterations, which is about the same as for the ring network of only eight links. This suggests that the convergence rate of the DCLS algorithm is not affected by the size of the network, which is a very important property.

3.3. *Large quasiregular network*

To illustrate the performance of the DCLS algorithm for an even larger network, we have considered the quasiregular network shown in Figure 6. The average distance between adjacent nodes is one. The upper-left plot shows the original set of 200 links, and the other five plots show five subsets of links determined by the DCLS algorithm with $K = 5$ and $\lambda = 40$. The partitions are surprisingly good. For $K = 2, 3, 4, 5, 6$, we have found that the corresponding values of λ_s are approximately 8, 16, 32, 40, 55.

Shown in Figure 7 is the convergence behavior of the DCLS algorithm which converged after about 30 iterations.

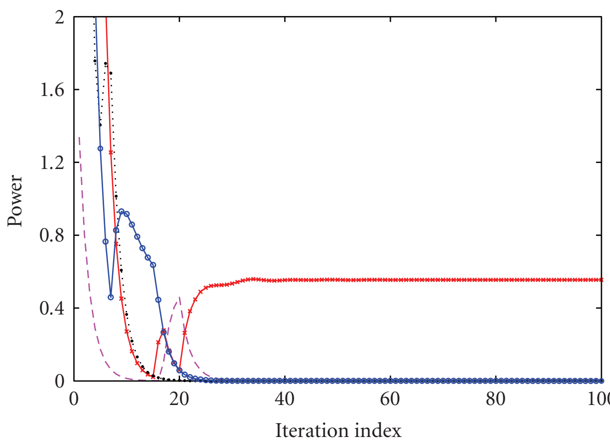

FIGURE 5: Illustration of $P_l^{(i)}(k)$ of the DCLS algorithm with $K = 4$ for the square network of 72 links versus the iteration index i, for an arbitrary l and all $k = 1, 2, 3, 4$, where $\lambda = 35$ and $\xi = 0.6$. At convergence, only one of $P_l^{(i)}(k)$ for $k = 1, 2, 3, 4$ is nonzero.

This once again suggests that the convergence rate of DCLS algorithm is not affected by the network size.

Shown in Figure 8 is the total transmission power consumed by the network, as determined by the DCPC algorithm following the DCLS algorithm, versus the values of K and for different values of the uniform link data rate $r_l = R$ in bits/s/Hz. We see that for this network and the chosen range of data rates $0.2 \le R \le 0.4$ bits/s/Hz, the optimal choice of K in terms of minimum power consumption is five.

By careful examination of each of the five subnetworks shown in Figure 6 for $K = 5$, we notice that almost all transmitters are two or more hops away from each receiver. This is an interesting validation of the two-hop rule in MSH-DSCH of IEEE 802.16.

But the DCLS algorithm is adaptive to the actual propagation environment, where the spacing between concurrent co-channel transmissions is established via distributed cooperative environmental sensing and calibration. Furthermore, different sparseness in different parts of the network can be achieved by choosing different values of λ_l at different links. The desired sparseness of a region should also be governed by the desired data rates in that region.

For very low data rates such as $R = 0.2$ bits/s/Hz, the difference among $K = 2, 3, 4, 5$ is not large. In this case, $K = 2$ should be a better choice than $K = 3, 4, 5$ because the peak data rate and peak power consumption for $K = 2$ are lower than those for $K = 3, 4, 5$.

We have tested the DCLS algorithm under various other conditions. The overall observations of the DCLS algorithm can be summarized as follows. (a) For any given K, there is λ_s such that when $\lambda \ge \lambda_s$, each link is scheduled to be on for only one of the total K time slots, and off for all other time slots. (b) For any given λ, there is K_s such that when $K \ge K_s$, the sparseness of each of the total K subnetworks remains about the same. (c) The convergence rate of the DCLS algorithm with a given ξ is not affected by the network size.

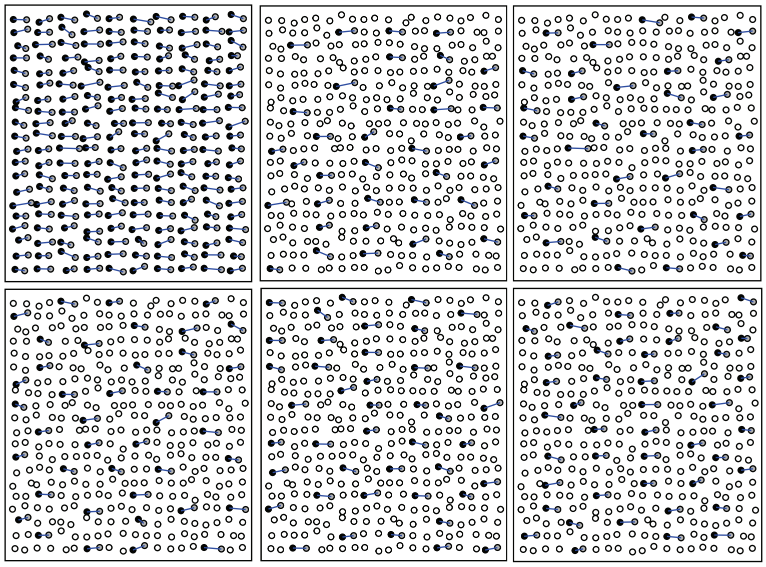

Figure 6: A quasiregular network of 200 links (upper left) is partitioned by the DCLS algorithm with $K = 5$ and $\lambda = 40$ into five subnetworks where the black circles represent the transmitting nodes and the grey circles the receiving nodes.

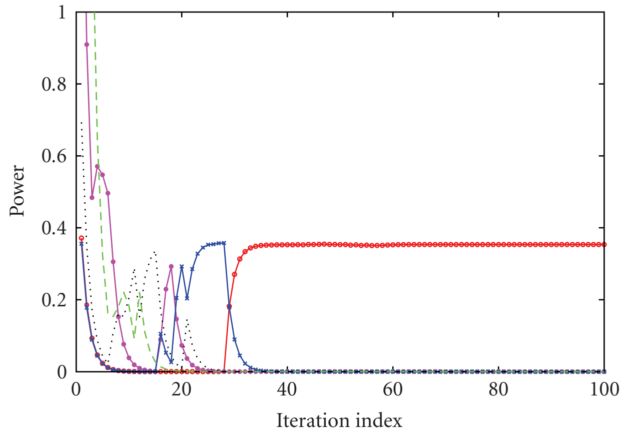

Figure 7: Illustration of $P_l^{(i)}(k)$ of the DCLS algorithm with $K = 5$ for the quasiregular network of 200 links versus the iteration index i, for an arbitrary l and all $k = 1, 2, 3, 4, 5$, where $\lambda = 32$ and $\xi = 0.5$. At convergence, only one of $P_l^{(i)}(k)$ for $k = 1, 2, 3, 4, 5$ is nonzero.

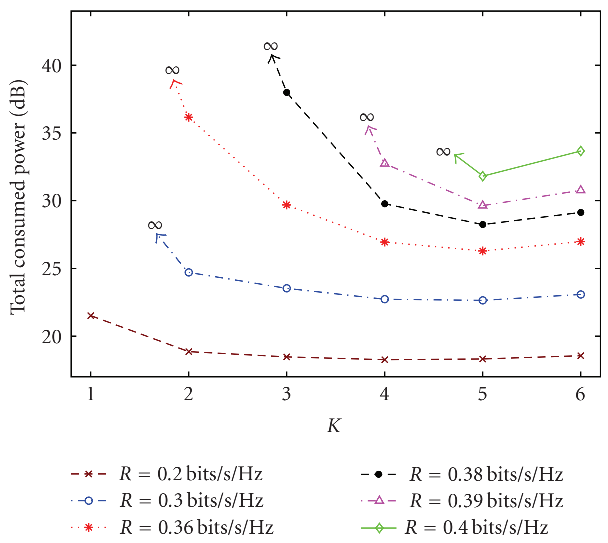

Figure 8: Total transmission power (dB) consumed by the quasiregular network versus K and for different uniform link data rates $r_l = R$ in bits/s/Hz, where ∞ denotes that no feasible solution was found by the DCPC algorithm following the DCLS algorithm.

4. CONCLUSIONS

We have presented a distributed and cooperative link scheduling (DCLS) algorithm which is especially useful for large-scale multihop wireless networks. This algorithm partitions a set of links into several subsets of links, where the sparseness of each subset is controlled by the parameters K (the number of time slots per frame) and λ (related to SINR margin). The convergence rate of the DCLS algorithm is apparently invariant to the network size, which is highly desirable for large-scale networks. Because of the spacing control provided by the DCLS algorithm, the total transmission power consumption as determined by the distributed and cooperative power control (DCPC) algorithm [7–9] can be significantly reduced to satisfy a given set of data rates of all links in the network. This property also translates to an increased feasible region of averaged link data rates.

Although verified through simulations, the convergence property of the DCLS algorithm remains to be established mathematically, which is an important future work. Practical implementation of the DCLS algorithm along with the DCPC algorithm is another interesting topic. Whether or not the DCLS algorithm can outperform MSH-DSCH as in IEEE 802.16 in a practical setting remains an important question.

ACKNOWLEDGMENTS

This work was supported in part by the U. S. National Science Foundation under Grants no. ECS-0401310 and TF-0514736, the U. S. Army Research Office under the MURI Grant no. W911NF-04-1-0224, and the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program.

REFERENCES

[1] R. L. Cruz and A. V. Santhanam, "Optimal routing, link scheduling and power control in multi-hop wireless networks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '03)*, vol. 1, pp. 702–711, 2003.

[2] H. Viswanathan and S. Mukherjee, "Throughput-range tradeoff of wireless mesh bachhaul networks," *IEEE Journal on Selected areas in Communications*, vol. 24, no. 3, pp. 593–602, 2006.

[3] J. Tang, G. Xue, C. Chandler, and W. Zhang, "Link scheduling with power control for throughput enhancement in multihop wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 55, no. 3, pp. 733–742, 2006.

[4] R. Bhatia and M. Kodialam, "On power efficient communication over multi-hop wireless networks: joint routing scheduling and power control," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '04)*, vol. 2, pp. 1457–1466, Hong kong, March 2004.

[5] J. Yuan, Z. Li, W. Yu, and B. Li, "A cross-layer optimization framework for multihop multicast in wireless mesh networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 11, pp. 2092–2102, 2006.

[6] C. S. R. Murthy and B. S. Manoj, *Ad Hoc Wireless Networks—Architectures and Protocols*, Prentice-Hall, Englewood Cliffs, NJ, USA, 2005.

[7] T. ElBatt and A. Ephremides, "Joint scheduling and power control for wireless ad hoc networks," *IEEE Transactions on Wireless Communications*, vol. 3, no. 1, pp. 74–85, 2004.

[8] G. J. Foschini and Z. Miljanic, "Simple distributed autonomous power control algorithm and its convergence," *IEEE Transactions on Vehicular Technology*, vol. 42, no. 4, pp. 641–646, 1993.

[9] R. D. Yates, "Framework for uplink power control in cellular radio systems," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 7, pp. 1341–1347, 1995.

[10] Y. Rong and Y. Hua, "Optimal power schedule for distributed MIMO links," in *Proceedings of the Army Science Conference*, Orlando, Fla, USA, November 2006.

[11] N. Abramson, "The ALOHA system—another alternative for computer communications," in *Proceedings of the AFIPS Fall Joint Computer Conference*, vol. 37, pp. 281–285, 1970.

[12] S. Xu and T. Saadawi, "Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks?" *IEEE Communication Magazine*, vol. 39, no. 6, pp. 130–137, 2001.

[13] M. Cao, W. Ma, Q. Zhang, and X. Wang, "Analysis of IEEE 802.16 mesh mode scheduler performance," *IEEE Transactions on Wireless Communications*, vol. 6, no. 4, pp. 1455–1464, 2007.

[14] Y. Hua, Y. Huang, and J. Garcia-Luna-Aceves, "Maximizing the throughput of large ad hoc wireless networks," *IEEE Signal Processing Magazine*, vol. 23, no. 5, pp. 84–94, 2006.

[15] K. Hong and Y. Hua, "Throughput analysis of large wireless networks with regular topologies," *EURASIP Journal on Wireless Communications and Networking*, vol. 2007, Article ID 26760, 11 pages, 2007.

[16] Y.-H. Lin, T. Javidi, R. L. Cruz, and L. B. Milstein, "Distributed link scheduling, power control and routing for multi-hop wireless MIMO networks," in *Proceedings of the Fortieth Asilomar Conference on Signals, Systems and Computers (ACSSC '06)*, pp. 122–126, Pacific Grove, Calif, USA, October-November 2006.

[17] K. Wang, C. F. Chiasserini, R. R. Rao, and J. G. Proakis, "A distributed joint scheduling and power control algorithm for multicasting in wirelss ad-hoc networks," in *Proceedings of the IEEE International Conference on Communications (ICC '03)*, vol. 1, pp. 725–731, Anchorage, Alaska, USA, May 2003.

[18] W. Wang, Y. Wang, X. Li, W. Song, and O. Frieder, "Efficient interference-aware TDMA link scheduling for static wireless networks," in *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking (MOBICOM '06)*, Los Angeles, Calif, USA,, September 2006.

[19] S. Gandham, M. Dawande, and R. Prakash, "Link scheduling in sensor networks: distributed edge coloring revisited," in *Proceedings of the IEEE 24th Annual Joint Conference of Computer and Communications Societies (INFOCOM '05)*, vol. 4, pp. 2492–2501, March 2005.