Distributed Cross-Layer Optimization for Cognitive Radio Networks

Yi Shi, *Member, IEEE*, Y. Thomas Hou, *Senior Member, IEEE*, Huaibei (Barry) Zhou, and Scott F. Midkiff, *Senior Member, IEEE*

***Abstract*—This paper presents a distributed cross-layer optimization algorithm for a multihop cognitive radio network, with the objective of maximizing data rates for a set of user communication sessions. We study this problem with joint consideration of power control, scheduling, and routing. Even under a centralized approach, such a problem has a mixed-integer nonlinear program formulation and is likely NP-hard. Thus, a distributed problem is very challenging. The main contribution of this paper is the development of a distributed optimization algorithm that iteratively increases data rates for user communication sessions. During each iteration, our algorithm has routing, minimalist scheduling, and power control/scheduling modules for improving the current solution at all three layers. To evaluate the performance of the distributed optimization algorithm, we compare it with an upper bound of the objective function. Results show that the distributed optimization algorithm can achieve a performance close to this upper bound. Because the optimal solution (unknown) is between the upper bound and the solution obtained by our distributed algorithm, we conclude that the results obtained by our distributed algorithm are highly competitive.**

***Index Terms*—Cognitive radio network (CRN), cross-layer optimization, distributed algorithm, power control, routing, scheduling.**

I. Introduction

COGNITIVE RADIO (CR) is a core technology for next-generation wireless networks. In a CR network (CRN), each node is equipped with a CR for wireless communications, which employs recent advances in RF design, signal processing, and communications software [26]. Such a node can dynamically access spectrum, and thus, a CRN has great potential to improve spectrum efficiency. In addition to the well-known primary/secondary network setting, CR's capability of sensing, adaptation, and learning makes it a candidate for many other important applications [26]. For example, CR is the key technology for radio interoperability in the U.S. military, i.e., the Joint Tactical Radio System (JTRS) program [9], public safety, i.e., SAFECOM program [19], and future mobile base stations, e.g., see the product line from Vanu Inc. [25].

There are some unique features in a CRN. A node may have a different set of available frequency bands, each may be of unequal size. Due to the software nature of CR, a node can simultaneously use multiple available frequency bands. From the wireless networking perspective, these new features of CRNs offer a whole new set of research problems in algorithm design and protocol implementation.

In this paper, we consider how we can design a distributed optimization algorithm for a CRN. The goal is to optimize network resource utilization, with the specific objective of maximizing a global scaling factor K under a given minimum rate requirement $r(l)$ for each session l. That is, we aim at finding the maximum K such that at least $K \cdot r(l)$ amount of data can be transported for each session l. Because power control directly affects the receiving power at a destination node (signal power) and at other nodes (interference power), it has a profound impact on interference relationship among the nodes and on scheduling. Moreover, power control and scheduling determine link capacities, which, in turn, affect routing. Thus, a networking problem for CRNs is inherently cross layer in nature and calls for joint consideration of power control, scheduling, and routing.

The main contribution of this paper is the development of a distributed optimization algorithm. We give details of the iterative steps in the algorithm on how we can increase a scaling factor for user communication sessions. During each iteration, we aim at increasing the smallest scaling factor among all the sessions in the network. In our proposed algorithm, there are two separate processes: 1) a *Conservative Iterative Process* (CIP) and 2) an *Aggressive Iterative Process* (AIP). CIP aims at increasing the smallest scaling factor without affecting any other session, whereas under AIP, other sessions' scaling factors can be decreased, as long as the affected scaling factors do not fall below the one that is being increased. The need of AIP is easy to understand. The reason that CIP is needed is not trivial and will be explained in Section III-B.

Both CIP and AIP incorporate three modules: 1) routing; 2) minimalist scheduling; and 3) power control/scheduling. In the routing module, link cost can be defined by the so-called *bandwidth-footprint product* (BFP) metric [22]. In the minimalist scheduling module, scheduling assignments along the minimum-cost route are made only when there is no other choice (and thus follows minimalist approach). The reason for this minimalist approach is that power control may change the

Manuscript received January 24, 2010; revised May 8, 2010 and June 28, 2010; accepted July 1, 2010. Date of publication July 15, 2010; date of current version October 20, 2010. The work of Y. T. Hou and Y. Shi was supported in part by the National Science Foundation under Grant CNS-0721570. The work of H. Zhou was supported in part by the National Basic Research Program of China (973 Program) under Grant 2009CB3204001. The review of this paper was coordinated by Dr. Y.-C. Liang.

Y. Shi, Y. T. Hou, and S. F. Midkiff are with the Bradley Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061 USA (e-mail: yshi@vt.edu; thou@vt.edu; midkiff@vt.edu).

H. Zhou is with the International School of Software Engineering, Wuhan University, Wuhan 430072, China (e-mail: bzhou@whu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2010.2058875

conflict relationship among links. Therefore, scheduling assignment is best done with joint consideration of power control. Finally, the power control/scheduling module determines all the remaining scheduling assignments and transmission powers and increases flow rate on the minimum-cost route.

To evaluate the performance of the distributed optimization algorithm, we compare it to an upper bound of the objective function, because the optimal solution cannot be obtained by its mixed-integer nonlinear program (MINLP) formulation. Simulation results show that the achievable performance by our distributed algorithm is close to the upper bound. Because the optimal solution (unknown) lies between the upper bound and the feasible solution obtained by our distributed algorithm, we conclude that the results obtained by our distributed algorithm are very close to the optimal solution and thus are highly competitive.

The rest of this paper is organized as follows. In Section II, we present a mathematical model for power control, scheduling, and routing. In Section III, we give details on the design of our distributed optimization algorithm. In Section IV, we present simulation results for the distributed algorithm and compare the results with those from an upper bound of the objective function. Section V reviews related work, and Section VI concludes this paper.

II. Cognitive Radio Network Modeling

We consider a multihop CRN with a set of $\mathcal{N}$ nodes. We assume that each CR can work on a set of frequency bands (denoted as $\mathcal{M}$), and the bandwidth of each band is W.[1] As discussed in the Introduction, due to other users' activities, a node in CRN may not be allowed to use all bands in $\mathcal{M}$. Instead, a node performs spectrum sensing to identify the so-called spectrum white space[2] (or holes) and uses them for wireless communications. That is, each node $i \in \mathcal{N}$ identifies a set of available frequency bands $\mathcal{M}_i$ (the set of spectrum white space), which may not be the same at different nodes. We consider a set of $\mathcal{L}$ user communication (unicast) sessions, each with a minimum rate requirement. Denote $s(l)$ and $d(l)$ as the source and destination nodes of session $l \in \mathcal{L}$ and $r(l)$ as the minimum rate requirement of session l. In this paper, we want to find the maximum global scaling factor K such that at least $K \cdot r(l)$ amount of data can be transported for each session $l \in \mathcal{L}$.

A. Scheduling and Power Control

Scheduling for transmission at each node in the network can be done in time, frequency, or code domain. In this paper, we consider scheduling in the frequency domain in the form of frequency bands. We assume that there are a sufficient number of frequency bands for scheduling. If the number of bands is small at the beginning, then subband division should be performed so that a sufficient number of new subbands are available before running our algorithm. We now present the necessary and sufficient condition for successful transmission.

Suppose that band m is available at both nodes i and j, i.e., $m \in \mathcal{M}_i$ and $m \in \mathcal{M}_j$. Denote g_{ij}^m as the propagation gain for transmission from node i to node j on frequency band m and p_{ij}^m as the power for this transmission. Under the protocol interference model, a data transmission from node i to node j on frequency band m is successful only if the received transmission power at node j exceeds a power threshold P_T^m [22], i.e., $p_{ij}^m \cdot g_{ij}^m \geq P_T^m$. Based on this condition, we can calculate the minimum required transmission power at node i on band m as $(p_{ij}^m)_T = P_T^m / g_{ij}^m$. Then, we have

$$p_{ij}^m \geq \left(P_{ij}^m\right)_T . \tag{1}$$

In addition, under the protocol interference model, an interference power is considered nonnegligible only if it exceeds a threshold P_I^m at a receiver [22]. Suppose that there is a transmission from node i to node j on band m and another transmission $k \rightarrow h$ on the same band. Then, the interference from node k to node j is considered negligible only if $p_{kh}^m \cdot g_{kj}^m \leq P_I^m$. Based on this condition, we can calculate the maximum allowed transmission power at node k on band m (which is considered as interference at node j) as $(P_{kj}^m)_I = P_I^m / g_{kj}^m$. Then, to make the interference negligible on node j, we must have

$$p_{kh}^m \leq \left(P_{kj}^m\right)_I . \tag{2}$$

Both (1) and (2), when considered in isolation, are necessary conditions for successful transmission. However, when jointly considered, they become a sufficient condition.

We now formulate these conditions for successful transmission in a multihop network setting, with joint consideration of power control and scheduling. First, we introduce binary indicator x_{ij}^m, which is 1 if node i transmits data to node j on frequency band m; otherwise, the binary indicator is 0. As aforementioned, we consider scheduling in the frequency domain, and thus, once a band $m \in \mathcal{M}_i$ is used by node i for transmission or reception, this band cannot be used again by node i for other transmission or reception. That is, we have

$$\sum_{j \in \mathcal{T}_i^m} x_{ij}^m + \sum_{k \in \mathcal{T}_i^m} x_{ki}^m \leq 1 \tag{3}$$

where $\mathcal{T}_i^m$ is the set of nodes to which node i can transmit under full power $P_{\max}^m$ on band m, i.e., $\mathcal{T}_i^m = \{j : (p_{ij}^m)_T \leq P_{\max}^m, j \neq i, m \in \mathcal{M}_j\}$.[3] Similarly, denote $\mathcal{I}_j^m$ as the set of nodes that can produce interference on node j on band m under full power $P_{\max}^m$, i.e., $\mathcal{I}_j^m = \{k : (P_{kj}^m)_I \leq P_{\max}^m, m \in \mathcal{M}_k\}$. Based on (1) and (2), we have the following

[1] The case of heterogeneous bandwidth for each frequency band can easily be extended.

[2] In [14], a band is considered white space if it remains unoccupied for at least 10 min.

[3] We assume that $g_{ij}^m = g_{ji}^m$. As a result, $\mathcal{T}_i^m$ is also the set of nodes from which node i can receive under full power $P_{\max}^m$ on band m.

conditions for successful transmission on link $i \rightarrow j$ and interfering link $k \rightarrow h$:

$$p_{ij}^m \begin{cases} \in \left[\left(P_{ij}^m\right)_T, P_{\max}^m\right], & \text{if } x_{ij}^m = 1 \\ = 0, & \text{if } x_{ij}^m = 0 \end{cases}$$

$$p_{kh}^m \leq \begin{cases} \left(P_{kj}^m\right)_I, & \text{if } x_{ij}^m = 1 \\ P_{\max}^m, & \text{if } x_{ij}^m = 0 \end{cases} \quad \left(k \in \mathcal{I}_j^m, k \neq i, h \in \mathcal{T}_k^m\right).$$

Mathematically, these conditions can be rewritten as

$$p_{ij}^m \in \left[\left(P_{ij}^m\right)_T x_{ij}^m, P_{\max}^m x_{ij}^m\right] \tag{4}$$

$$p_{kh}^m \leq P_{\max}^m - \left(P_{\max}^m - \left(P_{kj}^m\right)_I\right) x_{ij}^m \quad \left(k \in \mathcal{I}_j^m - \{i\}, h \in \mathcal{T}_k^m\right). \tag{5}$$

Denote $P_{\max}$ as the maximum total transmission power at a node on all bands. Then, for each node i, we have

$$\sum_{m \in \mathcal{M}_i} \sum_{j \in \mathcal{T}_i^m} p_{ij}^m \leq P_{\max}. \tag{6}$$

B. Flow Routing Under Link Capacity Constraint

Recall that our objective is to maximize a global scaling factor K so that at least $K \cdot r(l)$ amount of data can be transported for each session $l \in \mathcal{L}$. That is, suppose that each session l has a scaling factor $K(l)$ so that an amount of $K(l) \cdot r(l)$ data can be transported for session l. We have

$$K \leq K(l) \quad (l \in \mathcal{L}). \tag{7}$$

Then, we want to maximize K by optimizing all $K(l)$, $l \in \mathcal{L}$. Due to the limited transmission range of a node, it is necessary to employ multihop for data routing. Furthermore, to achieve optimality, it is also necessary to allow flow splitting due to its ability for load balancing and increased flexibility.

Mathematically, this case can be modeled as follows. Denote $f_{ij}(l)$ as the data rate on link $i \rightarrow j$ that is attributed to session l, where $i \in \mathcal{N}$, $j \in \mathcal{T}_i = \bigcup_{m \in \mathcal{M}_i} \mathcal{T}_i^m$. If node i is the source node $s(l)$ of session l, then

$$\sum_{j \in \mathcal{T}_i} f_{ij}(l) = K(l) \cdot r(l). \tag{8}$$

If node i is the destination node $d(l)$ of session l, then

$$\sum_{k \in \mathcal{T}_i} f_{ki}(l) = K(l) \cdot r(l). \tag{9}$$

If node i is an intermediate relay node for session l, i.e., $i \neq s(l)$ and $i \neq d(l)$, then

$$\sum_{j \in \mathcal{T}_i}^{j \neq s(l)} f_{ij}(l) = \sum_{k \in \mathcal{T}_i}^{k \neq d(l)} f_{ki}(l). \tag{10}$$

In addition to the aforementioned flow-balance equations at each node $i \in \mathcal{N}$ for session $l \in \mathcal{L}$, the aggregated flow rates on each radio link cannot exceed this link's capacity, i.e.,

$$\sum_{l \in \mathcal{L}}^{s(l) \neq j, d(l) \neq i} f_{ij}(l) \leq \sum_{m \in \mathcal{M}_{ij}} W \log_2\left(1 + \frac{g_{ij}^m}{\eta W} p_{ij}^m\right) \tag{11}$$

where η is the ambient Gaussian noise density. Note that the denominator inside the log function contains only ηW. This condition is due to the protocol interference model, i.e., when node i transmits to node j on band m, the interference from all other nodes in this band must be kept negligible to meet the scheduling constraint.

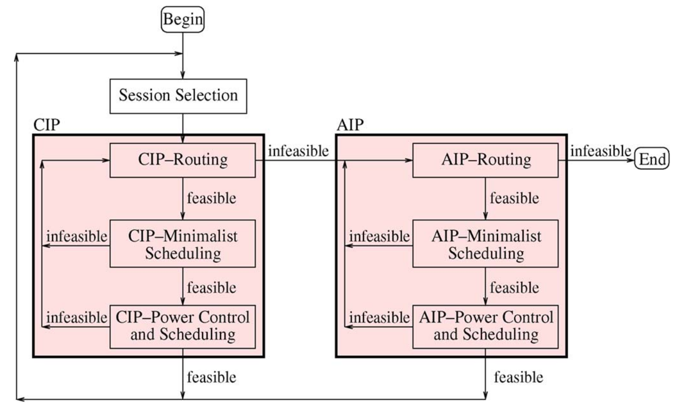

Fig. 1. Flow chart of our algorithm.

III. Design of a Distributed Optimization Algorithm

In this section, we present a distributed optimization algorithm for our problem. The main idea of this algorithm is given in Section III-A, which includes session selection, routing, minimalist scheduling, and power control/scheduling modules. The details of each module are described in Section III-B.

A. Main Idea

Our distributed algorithm iteratively increases the smallest scaling factor among all sessions and terminates when it cannot be further increased. In a distributed network, the source node $s(l)$ of each session $l \in \mathcal{L}$ maintains its own current scaling factor $K(l)$. Recall that, by (7), the objective K is equal to the smallest scaling factor among all sessions. During each iteration, source nodes elect the smallest scaling factor among all sessions in $\mathcal{L}$ by broadcasting their scaling factors [15]. Note that at different hops, these broadcast messages can be transported on different bands. Thus, there is no need to have a common band in the network for broadcast. When there are multiple sessions with the same smallest scaling factor, the tie is deterministically broken by choosing the session with the smallest source node ID.

Upon identifying the session l with the smallest current scaling factor $K(l)$, the algorithm moves onto an iteration process as shown in Fig. 1. There are two separate processes: CIP (see Fig. 1, left) and AIP (see Fig. 1, right). Although both processes contain routing, minimalist scheduling, and power control/scheduling modules, they differ in objectives and details. The objective of CIP is to increase $K(l)$ without affecting (decreasing) the current scaling factors of other sessions in $\mathcal{L}$. On the other hand, the objective of AIP is to aggressively increase $K(l)$ by decreasing the current scaling factors of some other sessions in $\mathcal{L}$, as long as they do not fall below the newly increased $K(l)$. The need of AIP is easy to understand. Therefore, we have the following question: Why is CIP necessary? The answer to this question lies in the way the link cost is defined in the AIP–Routing module, which is different from that in the CIP–Routing module. We will explain in detail how the definition of link cost in AIP–Routing module calls for the need of CIP in Section III-B (see Remark 1).

We now present the ideas in the routing module under CIP and AIP. During an iteration, the routing, scheduling, and power control for session l in the previous iterations are intact. The CIP–Routing module aims at finding an additional route (which could overlap with previous routes) for session l onto which there is a potential to push more data rate. This routing module is based on the minimum cost, which can distributedly be implemented. The key step in CIP–Routing is the definition of incremental link cost (ILC) for pushing more data rate onto a link. This cost should only require local information and can distributedly be computed. On the other hand, under the AIP–Routing module, all links that carry sessions whose current scaling factors are greater than $K(l)$ will be considered. The cost on these links will be redefined to reflect that session l may push more data rate at the expense of decreasing the data rate of sessions currently with larger scaling factors.

We now present the ideas in the minimalist scheduling module under CIP and AIP. Our approach is minimalist, because only necessary scheduling decisions are made. In particular, under CIP–Minimalist Scheduling, a new band should be assigned to a hop if there is no remaining capacity on this hop and current transmission powers on all active bands cannot be further increased. If there is only one unassigned band on this hop, the transmitter of this hop will use this band (because there is no other option), subject to the scheduling constraint (3). On the other hand, when there are multiple unassigned bands, the minimalist approach calls for deferring band assignment in the power control/scheduling module (see the following paragraph). The reason for this deferring is that power control may change the conflict relationship among the nodes. Therefore, scheduling decision (band assignment among multiple unassigned bands) is best done with joint consideration of power control. The AIP–Minimalist Scheduling module follows a similar process, with the difference being when a new band should be assigned. This condition is because, under AIP–Minimalist Scheduling, if a hop carries sessions with their current scaling factors greater than $K(l)$, then there is no need to assign a new band; the rates of these sessions can be reduced to leave more room for increasing the rate of session l.

Finally, we discuss the power control/scheduling module under CIP and AIP. This module sets the transmission power on a currently active band or some new band along the minimum-cost route chosen in the routing module. The objective is to allow some additional flow rate $f(l)$ to be transported on this route for session l. The specific value of $f(l)$ can be determined hop by hop along the minimum-cost route. Under CIP–Power Control/Scheduling, each transmitter along the route tries the following strategies to accommodate $f(l)$: 1) Use the remaining capacity on this hop if possible; 2) increase the transmission power on a currently active band to increase link capacity; and 3) use a new frequency band (previously unassigned). Recall that, in the minimalist scheduling approach, band assignment is deferred when there are multiple unassigned bands on a hop. This deferred band assignment is fulfilled in strategy 3. It is also important to realize that, when a new band is assigned, the maximum allowed transmission power at a nearby node on this band may be reduced by (5). The AIP–Power Control/Scheduling module is similar to the CIP–Power Control/Scheduling module, but it has one more strategy for accommodating $f(l)$. That is, in strategy 1, AIP–Power Control/Scheduling will also check whether $f(l)$ can be accommodated by the newly released capacity from sessions currently with larger scaling factors.

It should be clear that the smallest scaling factor among all the sessions will be increased after either CIP or AIP is successfully completed after an iteration.

B. Algorithm Details

Before we present the details in our algorithm, we first introduce the following notation. For the distributed algorithm, we redefine x_{ij}^m as follows:

$$x_{ij}^m = \begin{cases} 1, & \text{if band } m \text{ is used (assigned) on link } i \to j \\ 0, & \text{if band } m \text{ is unassigned on link } i \to j \\ -1, & \text{if band } m \text{ cannot be used on link } i \to j. \end{cases}$$

Another notation that we need in the iteration of the distributed algorithm is the maximum allowed transmission power $(p_{ij}^m)_U$. Recall that, by (4), $P_{\max}^m$ is the maximum transmission power at a node on band m. During an iteration, a node may find that, under (5) and (6), the current maximum allowed transmission power may be smaller than $P_{\max}^m$. We use $(p_{ij}^m)_U$ for this purpose, where subscript U indicates the current upper bound on the transmission power.

Routing Module: As discussed in Section III-A, the key step in the routing module is the definition of link cost that captures network resource usage. Network resource usage can be defined in a number of ways, which typically includes bandwidth usage. However, as pointed out in [13], bandwidth usage can only characterize network resource usage in spectrum, but not the impact (i.e., interference) of radio transmission in space. For example, a node that transmits with the same channel bandwidth but with different power levels will produce different interference footprint areas. To account for both the spectrum usage and the impact of a CR in space dimension, the so-called BFP metric was introduced in [22] and will also be adopted in this paper. Therefore, the ILC for pushing more data rate onto a link can be defined as the incremental BFP per additional data rate. For our problem, because each band has the same bandwidth, BFP reduces to footprint, which is the interference area for a transmission. Therefore, the definition of ILC becomes the additional footprint over the increase of flow rate for the session with the smallest scaling factor.

To compute the footprint area, we adopt the simple propagation gain model $g_{ij}^m = d_{ij}^{-\alpha_m}$, where d_{ij} is the physical distance between nodes i and j, and α_m is the path loss index on band m. For a transmission from node i to node j on band m, based on (5), a node h is interfered when $p_{ij}^m > (P_{ih}^m)_I = P_I^m / g_{ih}^m$. Thus, the interference range of node i on band m is $(p_{ij}^m / P_I^m)^{1/\alpha_m}$, and the footprint is $\pi (p_{ij}^m / P_I^m)^{2/\alpha_m}$.

Because a wireless link in a CRN is associated with multiple frequency bands, the computation of ILC must, somehow, be related to the cost of each frequency band. We give some details here on how ILC is computed. Initially, for each band m on link $i \rightarrow j$, because $p_{ij}^m = 0$ (zero transmission power), both the capacity on this band and the footprint area are 0. In our algorithm, when a new band is used, the transmission power is set to the minimum required transmission power $(P_{ij}^m)_T$, and further increase in the transmission power can only happen in future iterations. That is, we follow a conservative approach to determine the transmission power for a newly active band. Under this conservative approach, node i can compute the *incremental band cost* (IBC) for a newly active band m by

$$IBC(i,j,m) = \frac{\pi \left(\frac{(P_{ij}^m)_T}{P_I^m} \right)^{2/\alpha_m}}{W \log_2 \left(1 + \frac{g_{ij}^m (P_{ij}^m)_T}{\eta W} \right)}. \quad (12)$$

IBC in (12) is initially identical on all bands; therefore, ILC can also be defined by (12).

In subsequent iterations, the definition of ILC is case specific. For the simple case where link $i \rightarrow j$ has a positive remaining capacity, node i sets its ILC to 0, because this link can support additional flow rate without increasing its transmission power (or its footprint). For other cases (i.e., no remaining capacity), the computation of IBC depends on whether the band is currently used.

- *Case 1*. If band m is already used and $p_{ij}^m < (p_{ij}^m)_U$, then p_{ij}^m may be increased to $(p_{ij}^m)_U$. Node i computes IBC as

$$IBC(i,j,m) = \frac{\pi \left[\frac{(p_{ij}^m)_U}{P_I^m} \right]^{2/\alpha_m} - \pi \left(\frac{p_{ij}^m}{P_I^m} \right)^{2/\alpha_m}}{W \log_2 \left[1 + \frac{g_{ij}^m}{\eta W} (p_{ij}^m)_U \right] - W \log_2 \left(1 + \frac{g_{ij}^m}{\eta W} p_{ij}^m \right)}. \quad (13)$$

- *Case 2*. If band m is not yet used, then node i computes IBC by (12).
- *Case 3*. If band m is already fully utilized (i.e., $(p_{ij}^m)_U = p_{ij}^m$), then node i sets IBC to ∞, because the capacity on this band cannot be further increased.

On link $i \rightarrow j$, different bands may have different IBCs; therefore, we need to have a band selection policy to decide which band will be used, and subsequently, node i can define its ILC based on the chosen band. The key idea of our band selection policy is to use a band that is already active to its fullest extent before considering deploying a new band. Thus, when there exists a case-1 band, node i chooses such a band with the smallest IBC. As a result, ILC will be defined as the IBC of the chosen band. Otherwise, node i examines if there exists a case-2 band and will use it if available. As a result, ILC will be defined as the IBC of such a case-2 band. Note that case-3 bands are already fully utilized. When there are only case-3 bands, node i sets its ILC to ∞, because this link's capacity cannot be further increased. The pseudocode for ILC computation in CIP–Routing is given in Fig. 2.

```
Determine link cost for link i → j in CIP-Routing
if node i finds that the remaining capacity c_ij > 0
    Node i sets ILC(i, j) = 0;
else, if there is at least one band m with x_ij^m = 1 and p_ij^m < (p_ij^m)_U {
    Node i computes IBC for each of these bands by (13);
    Node i defines ILC(i, j) by the smallest IBC of these bands; }
else, if there is at least one band m with x_ij^m = 0
    Node i computes IBC(i, j, m) by (12) and sets ILC(i, j) =
    IBC(i, j, m);
else, node i sets ILC(i, j) = ∞;
```

Fig. 2. Link cost in CIP–Routing at a node i.

For the AIP–Routing module, there is an additional option of reducing flow rates of some sessions with larger scaling factors to increase the smallest scaling factor of a session under consideration. Note that, in this scenario, the smallest scaling factor is increased, whereas no transmission power (or footprint) is increased. Thus, node i sets its ILC to 0, because there is no change in footprint.

Under both CIP–Routing and AIP–Routing, the ILC of a link may be different during each iteration. As a result, the minimum-cost route at each iteration could be different for the same session. The union of these routes for all the iterations will lead to a multipath routing solution for a session, which is important in terms of maximizing the objective K.

Remark 1: With link cost definitions in CIP and AIP, we can now explain our earlier question of why CIP is needed in our algorithm (see Fig. 1). Note that, under AIP, any link that has sessions with larger scaling factors than the session under optimization will have zero link cost. If AIP is used alone, then many sessions may attempt to traverse such zero-cost links, making such links bottleneck in the network. By using CIP before AIP, we can more evenly distribute session rates among the network without getting into such a bottleneck situation. Therefore, CIP is an essential mechanism for the proper operation of the distributed optimization algorithm, whereas AIP is only used as an enhancement mechanism.

Minimalist Scheduling Module: As described in Section III-A, we follow a minimalist approach in scheduling (band assignment), i.e., a band is assigned only when there are no other alternatives.

Now, we describe how the minimalist scheduling is done along the minimum-cost route from a source to its destination (see Fig. 3). For each hop $i \rightarrow j$, if node i finds that there is no remaining capacity on hop $i \rightarrow j$ and its transmission power on all active bands cannot be increased, then it is necessary to assign a new band on this hop. In this case, if there is only one remaining band for assignment, then node i uses this band and sets $x_{ij}^m = 1$.

Upon the assignment of a new band on hop $i \rightarrow j$, it is necessary to make updates in both backward (toward the source) and forward (toward the destination) directions. In the backward direction, based on the scheduling constraint (3), this band must not be used on the previous hop, e.g., $k \rightarrow i$. That is, band m can be either unusable or unassigned on hop $k \rightarrow i$. The backward update procedure is done for the unusable case. For the case of the unassigned, node k needs to set band m to be

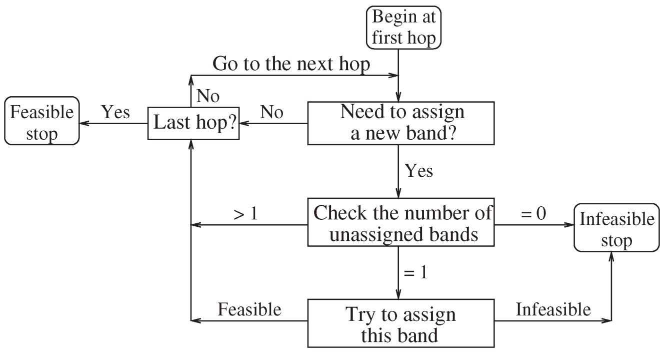

Fig. 3. Flow chart of the minimalist scheduling module.

CIP-Minimalist Scheduling

```
MinScheduling(int l) {
    From the first hop to the last hop in the minimum-cost route {
        Assume the current hop is i → j.
        if (CheckLink(i, j, l)==−1) {
            Node i sets ILC(i, j) = ∞;
            Revoke all scheduling assignments in this iteration, from
            the current node i to the source node s(l);
            Minimalist scheduling along the current route is infeasible.}}
    Minimalist scheduling along the current route is feasible.
}
//Node i returns 1 if feasible; −1 if infeasible
//mode = 1: need to check backward direction; mode = 2: need to
//check forward direction; mode = 3: need to check both directions
int CheckLink(int i, int j, int l, int mode = 3)
{
    if remaining capacity c_ij > 0, then node i returns 1.
    if there is at least one band m with x^m_ij = 1 and p^m_ij < (p^m_ij)_U,
    then node i returns 1.
    Node i counts the number of unassigned bands (with x^m_ij = 0);
    if there is no unassigned band, then node i returns −1.
    else, if there are multiple unassigned bands, then node i returns 1.
    else {
        Assume band m is the only unassigned band;
        Node i sets x^m_ij = 1;
        if ((mode%2==1)&&(i!=s(l))) {
            Suppose the previous hop is k → i;
            if (x^m_ki==0) {
                Node k sets x^m_ki = −1;
                if node k has not activated a new band on link k → i,
                then CheckLink(k, i, l, 1); } }
        if ((mode/2==1)&&(j!=d(l))) {
            Suppose the next hop is j → h;
            if (x^m_jh==0) {
                Node j sets x^m_jh = −1;
                if (CheckLink(j, h, l, 2)==−1) {
                    Node j informs node i that scheduling is infeasible.
                    Node i returns −1. } } }
        Node i returns 1. }
}
```

Fig. 4. Pseudocode for the CIP–Minimalist Scheduling module.

unusable, i.e., set $x_{ki}^m = -1$. Due to this operation (removal of band m on hop $k \rightarrow i$), node k may or may not activate a new band on hop $k \rightarrow i$ (following the same token in the last paragraph). If no assignment should be made on hop $k \rightarrow j$, then the backward update procedure is done. Otherwise, node k will make the assignment. Subsequently, after such a new band assignment, it is necessary to further go backward to make necessary updates.

In the forward direction, the update follows the same token, except that we may encounter an infeasible situation. In this case, link $i \rightarrow j$ should be removed from future minimum-cost routing by setting its ILC to ∞. Furthermore, all the scheduling assignments previously done will be revoked, from the current node i to the source node $s(l)$. The pseudocode for CIP–Minimalist Scheduling is given in Fig. 4.

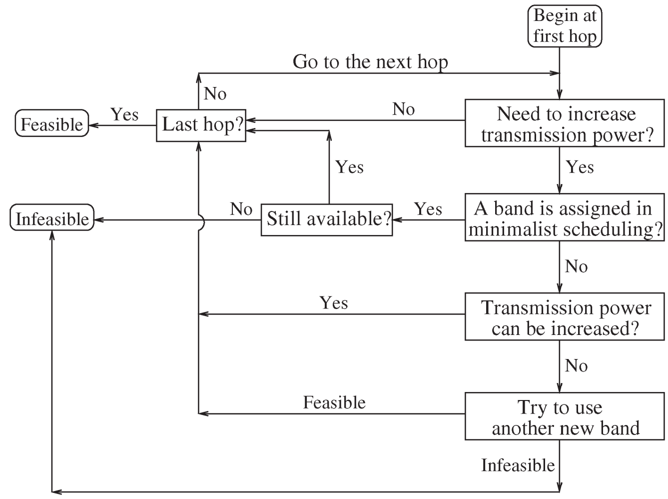

Fig. 5. Flow chart of the power control/scheduling module.

For the AIP–Minimalist Scheduling module, because there is an additional option of reducing the flow rate of some other sessions with larger scaling factors, the decision of when a new band must be assigned will thus differ from that in the CIP–Minimalist Scheduling module. Recall that, in CIP–Minimalist Scheduling, a new band should be assigned if there is no remaining capacity on a link and transmission power on all active bands cannot be increased. In contrast, in AIP–Minimalist Scheduling, a new band is activated only when the link does not have any other session with a larger scaling factor, in addition to those conditions in CIP–Minimalist Scheduling. The pseudocode for AIP–Minimalist Scheduling is similar to Fig. 4 and is omitted.

One important observation is that, during minimalist scheduling, only nodes that are within one hop away (under $P_{\max}^m$) along the minimum-cost route may be affected. The effect of assigning band m on a link $i \rightarrow j$ is that band m can no longer be used again by either node i or node j for transmission/reception. That is, for nodes that are one-hop neighbors of node i or j, band m can no longer be used for transmitting to or receiving from node i or j. Thus, the effect of scheduling assignment at node i on band m is limited to nodes that are within one hop from either node i or node j. By applying the same analysis to all the nodes along the route, it is not hard to see that only nodes that are within one hop away from the nodes along the minimum-cost route may be affected.

Power Control/Scheduling Module: The last module in either CIP or AIP is power control/scheduling (see Fig. 5). In this module, all the remaining scheduling assignments (that are not determined in the minimalist scheduling module), transmission powers, and flow rates on the minimum-cost route will be determined.

Again, power control/scheduling is performed on a hop-by-hop basis along the route from the source to the destination. The potential increase $f(l)$ in session flow rate can also be computed hop by hop. Under CIP–Power Control/Scheduling, this method is accomplished with the following steps.

Step 1) For hop $i \rightarrow j$, node i checks whether there is a positive remaining capacity c_{ij}. If yes, then node i updates the flow rate increase $f(l)$ by $\min\{c_{ij}, f(l)\}$

CIP-Power Control/Scheduling

```
PowerControl(int l) {
    Source node s(l) initializes f(l) = ∞;
    From the first hop to the last hop in the minimum-cost route {
        Assume the current hop is i → j.
        For each band m with x_{ij}^m changed from 0 to −1 in
        MinScheduling(), node i sets (p_{ij}^m)_U = 0;
        if (c_{ij} > 0) {
            Node i sets f(l) = min{c_{ij}, f(l)};
            Power control and scheduling are feasible for this hop. }
        if there is a band m with x_{ij}^m changed from 0 to 1 in
        MinScheduling() {
            if (Active(i, j, m, f(l))==−1) {
                Node i sets ILC(i, j) = ∞;
                Revoke all power control/scheduling decisions done in this
                iteration, from the current node i to the source node s(l);
                Flow rate along the current route cannot be increased. }
            Power control and scheduling are feasible for this hop. }
        if there is a band with x_{ij}^m = 1 and p_{ij}^m < (p_{ij}^m)_U {
            Suppose band m has smallest IBC among these bands;
            if the increased capacity under (p_{ij}^m)_U is less than f(l)
                Node i computes the increased capacity and updates f(l)
                by this capacity;
            Power control and scheduling are feasible for this hop. }
        Node i sets found = 0;
        Node i examines each band with x_{ij}^m = 0 in the non-decreasing
        order of (p_{ij}^m)_U {
            if (Active(i, j, m, f(l))==1), {
                Node i sets found = 1;
                Power control and scheduling are feasible for this hop. }}
        if (found==0) {
            Node i sets ILC(i, j) = ∞;
            Revoke all power control/scheduling decisions done in this
            iteration, from the current node i to the source node s(l);
            The flow rate along the current route cannot be increased. } }
    From the first hop to the last hop in the minimum-cost route {
        Assume the current hop is i → j.
        Node i sets f_{ij}(l) = f_{ij}(l) + f(l);
        if (c_{ij} > 0) {
            Node i sets c_{ij} = c_{ij} − f(l);
            Power control and scheduling are done for this hop. }
        if band m is chosen among bands with x_{ij}^m = 1 and
        p_{ij}^m < (p_{ij}^m)_U {
            Node i increases the transmission power to a suitable value to
            support f(l);
            Node i updates the maximum allowed transmission power to
            other node on other band by (6);
            Power control and scheduling are done for this hop. }
        if band m is chosen among bands with x_{ij}^m = 0 {
            Node i sets x_{ij}^m = 1, p_{ij}^m = (P_{ij}^m)_T, and computes the
            remaining capacity c_{ij}; } }
    Source node s(l) updates K(l).
}
```

Fig. 6. Pseudocode for the CIP–Power Control/Scheduling module.

```
//If link i → j can be active on band m, node i returns 1, else returns −1
int Active(int i, int j, int m, double f(l))
{
    For each k ∈ T_i^m {
        if (p_{ki}^m > 0) {
            Node k informs node i that link i → j cannot be active on
            band m.
            Node i returns −1. }
        Node k sets x_{ki}^m = −1 and (p_{ki}^m)_U = 0; }
    For each h ∈ T_i^m and h ≠ j {
        if (p_{ih}^m > 0), then node i returns −1.
        Node i sets x_{ih}^m = −1 and (p_{ih}^m)_U = 0; }
    For each k ∈ T_j^m and k ≠ i {
        if (p_{kj}^m > 0) {
            Node k generates a message of "link i → j cannot be active
            on band m " and lets node j forward this message to node i.
            Node i returns −1. }
        Node k sets x_{kj}^m = −1 and (p_{kj}^m)_U = 0; }
    For each h ∈ T_j^m {
        if (p_{jh}^m > 0) {
            Node j informs node i that link i → j cannot be active on
            band m.
            Node i returns −1. }
        Node j sets x_{jh}^m = −1 and (p_{jh}^m)_U = 0; }
    For each k ∈ I_j^m, k ≠ i, and h ∈ T_k^m
        if ((P_{kj}^m)_I < (p_{kh}^m)_U) {
            By (5), node k sets (p_{kh}^m)_U = (P_{kj}^m)_I;
            if (p_{kh}^m > (p_{kh}^m)_U) {
                Node k generates a message of "link i → j cannot be
                active on band m" and lets node j forward this message
                to node i.
                Node i returns −1. }
            if ((p_{kh}^m)_U < (p_{kh}^m)_T)
                Node k sets (p_{kh}^m)_U = 0 and x_{kh}^m = −1;
    Node i sets p_{ij}^m = (P_{ij}^m)_T;
    if the increased capacity is smaller than flow rate f(l), then node i
    updates f(l) by the increased capacity;
    Node i returns 1.
}
```

Fig. 7. Pseudocode for the auxiliary function in the CIP–Power Control/Scheduling module.

to ensure (10) and (11) and completes the power control and scheduling on this hop; otherwise, it continues to step 2).

Step 2) Node i checks whether a band m is assigned in the minimalist scheduling module. If yes, then node i performs step 2.1) or 2.2); otherwise, it continues to step 3).

1) If band m is still available (after power control operation along the route), then node i uses the transmission power $(P_{ij}^m)_T$. If the newly achieved capacity is less than $f(l)$, node i needs to update $f(l)$ by this capacity. The power control and scheduling on this hop is completed.

2) If this band is no longer available, then node i detects an infeasible situation, because in the minimalist scheduling module, a band is assigned only if there is no other band available. In this case, hop $i \rightarrow j$ should be removed from future minimum-cost routing by setting its ILC to ∞. Furthermore, all the power control/scheduling previously done will be revoked, from the current node i to the source node $s(l)$.

Step 3) Node i checks whether it is possible to increase the transmission power on bands that are already used.

- If yes, then node i chooses a band m among such bands with the smallest IBC. On band m, node i will increase its transmission power to a level that can support the current $f(l)$ or the maximum allowed transmission power $(p_{ij}^m)_U$, whichever is smaller. In the case that the maximum allowed transmission power is used, node i also needs to update $f(l)$ by the newly achieved capacity. Because node i increases its transmission power to node j on band m, the maximum allowed transmission power to other nodes on other bands may be decreased due to (6). The power control and scheduling on this hop is completed.
- If no, then node i continues to step 4).

Step 4) Node i checks whether a new band whose maximum allowed transmission power is the largest among bands that are still available for assignment at this point. Subsequently, either step 2.1) (if an available band is found) or 2.2) (if no available band) will be performed by node i.

Once $f(l)$, the maximum achievable increase on flow rate, is determined at the last hop on the minimum-cost route, power control and scheduling setting will be updated to support this increase. Such an update is again performed along the minimum-cost route, from the first hop to the last hop. Then, source node $s(l)$ can increase $K(l)$ by (8). The pseudocode for CIP–Power Control/Scheduling is given in Figs. 6 and 7.

For the AIP–Power Control/Scheduling module, there is one more strategy for exploring to accommodate the additional flow rate $f(l)$. That is, in step 1), if there are other sessions with larger scaling factors on this link, then node i can obtain some additional capacity by reducing the scaling factor of one of these sessions. Among these sessions, the session with the largest releasable capacity is chosen. For this session, its flow rate on other hops along its paths also needs to be reduced. The transmission power and scheduling on these links may also need to be updated. The pseudocode for the main function of AIP–Power Control/Scheduling is similar to Fig. 6 and is omitted.

C. Complexity

We now show that our algorithm has a polynomial complexity by analyzing the complexity of each iteration and the total number of iterations. To analyze the complexity of each iteration, we need to analyze the complexity of each module in the algorithm, which can be measured by its communication overhead. Session selection can be performed by letting each source node broadcast its session information. Note that such a broadcast can be piggybacked in that for routing. Thus, there is no extra overhead for session selection. The complexity for each of the three modules is analyzed as follows.

- For both CIP–Routing and AIP–Routing, the minimum-cost routing can be done by letting each node broadcast its costs to each of its one-hop neighbors. Then, the source node of the chosen session can compute the minimum-cost route. The complexity for a broadcast is $O(|\mathcal{N}|^2)$.
- The complexity of CIP–Minimalist Scheduling depends on the complexity of CheckLink() at each hop (see Fig. 4). Because a link's upstream or downstream link is checked only if a new band is set to be active on this link, the complexity at each hop is on the order of the number of links that a new band is set to be active in this CheckLink(). Note that a new band can be set at most once on each link in MinScheduling() and the number of links in the minimum-cost route is at most $|\mathcal{N}| - 1$. Thus, the total complexity of CheckLink() at each hop is $O(|\mathcal{N}|)$, which is also the complexity of CIP–Minimalist Scheduling. Because the number of links with a new band that is set to be active in AIP–Minimalist Scheduling is smaller than that in CIP–Minimalist Scheduling, the complexity of AIP–Minimalist Scheduling is also $O(|\mathcal{N}|)$.
- For CIP–Power Control/Scheduling, major computations are in the first iteration in PowerControl() (see Fig. 6). The complexity is the product of the number of hops (at most $|\mathcal{N}| - 1$), the number of unassigned bands (at most $|\mathcal{M}|$), and the complexity of Active(). The complexity of Active() is on the order of the number of interfering nodes, which is at most $O(|\mathcal{N}|)$. Therefore, the complexity of CIP–Power Control/Scheduling is $O(|\mathcal{N}|^2|\mathcal{M}|)$. The additional complexity in AIP–Power Control/Scheduling is on the order of the number of hops in the route for the chosen session l (at most $|\mathcal{N}| - 1$) times the complexity of released flow rates for another session (at most $O(|\mathcal{N}|^2)$). Therefore, the additional complexity is $O(|\mathcal{N}|^3)$, and the overall complexity of AIP–Power Control/Scheduling is $O(|\mathcal{N}|^2|\mathcal{M}| + |\mathcal{N}|^3)$.

Based on the aforementioned complexity analysis for each of the three modules, we can analyze the complexity of each iteration in our distributed algorithm. When CIP–Minimalist Scheduling or CIP–Power Control/Scheduling is infeasible, a link's ILC will be set to ∞. Then, with at most $O(|\mathcal{N}|^2)$ infeasible trials of CIP–Minimalist Scheduling or CIP–Power Control/Scheduling, we will find that either CIP–Routing is infeasible (i.e., the current CIP iteration ends with an infeasible result) or the current smallest scaling factor is increased (i.e., the current CIP iteration ends with a feasible result). Thus, the complexity of one CIP iteration is $[O(|\mathcal{N}|^2) + O(|\mathcal{N}|) + O(|\mathcal{N}|^2|\mathcal{M}|)] \cdot O(|\mathcal{N}|^2) = O(|\mathcal{N}|^4|\mathcal{M}|)$. Similarly, the complexity of one AIP iteration is $O(|\mathcal{N}|^4|\mathcal{M}| + |\mathcal{N}|^5)$.

We then analyze the number of total iterations in our distributed algorithm. During the convergence, each iteration can have one of the following four outcomes: 1) CIP–feasible; 2) CIP–infeasible; 3) AIP–feasible; and 4) AIP–infeasible. Note that the first iteration should yield a CIP–feasible result and the last two iterations should yield CIP–infeasible and AIP–infeasible results, respectively. For iterations with CIP–feasible, we can find a link $i \to j$ and a band m such that the transmission power p_{ij}^m is increased to either $(P_{ij}^m)_T$ or $(p_{ij}^m)_U$. Thus, the transmission power at node i on band m will be increased to the maximum allowed transmission power in two (may not be consecutive) CIP–feasible iterations. Therefore, the number of CIP–feasible iterations is at most $2|\mathcal{N}||\mathcal{M}|$.

Now, we need to analyze the number of iterations between two consecutive CIP–feasible iterations. There are equal numbers of CIP–infeasible and AIP–feasible iterations, with each CIP–infeasible iteration followed by an AIP–feasible iteration. If we examine a particular session's rate in AIP–feasible iterations, then we can find that its rate may increase multiple times (along with decreased rates for other sessions). Because each such increase will decrease a different session's rate, the number of AIP–feasible iterations is at most $|\mathcal{L}|^2$. Then, the number of iterations between two consecutive CIP–feasible iterations is at most $2|\mathcal{L}|^2$. The total number of iterations is no more than $2|\mathcal{N}||\mathcal{M}| \cdot (2|\mathcal{L}|^2 + 1) + 2 = O(|\mathcal{N}||\mathcal{M}||\mathcal{L}|^2)$.

Because the complexity of one iteration is at most $O(|\mathcal{N}|^4|\mathcal{M}| + |\mathcal{N}|^5)$ and the total number of iterations is no more than $O(|\mathcal{N}||\mathcal{M}||\mathcal{L}|^2)$, the overall complexity is $O(|\mathcal{N}||\mathcal{M}||\mathcal{L}|^2) \cdot O(|\mathcal{N}|^4|\mathcal{M}| + |\mathcal{N}|^5) = O(|\mathcal{N}|^5|\mathcal{M}|^2|\mathcal{L}|^2 + |\mathcal{N}|^6|\mathcal{M}||\mathcal{L}|^2)$. We point out that this result is an upper bound for the complexity and that the actual complexity will be much lower.

IV. Simulation Results

In this section, we present simulation results to demonstrate the performance of our distributed algorithm. Ideally, the best performance benchmark would be the optimal solution.

However, as shown in [20], the problem formulation is in the form of MINLP and is likely NP-hard. Because the optimal solution is not available, we will compare the performance of our distributed optimization algorithm to an upper bound developed in [20]. Simulation results show that the achievable objective by our distributed algorithm is close to the upper bound. Because the optimal solution (unknown) lies between the upper bound and the feasible solution obtained by our distributed algorithm, we conclude that the results obtained by our distributed algorithm are even closer to the optimal solution and are thus highly competitive.

A. Simulation Setting

In this paper, we consider 100 network instances. Each network has either $|\mathcal{N}| = 20, 30, 40, \text{or } 50$ nodes randomly deployed in a 100×100 area. The units for distance, rate, and power density are all normalized with appropriate dimensions. There are $|\mathcal{M}| = 10$ different frequency bands in the network. However, at each node, only a subset of these frequency bands may be available. In the simulation, this approach is done by randomly selecting a subset of bands for each node from a pool of ten bands. Each band has a bandwidth of $W = 50$. Among these nodes, there are $|\mathcal{L}| = 3$ or 5 sessions, with the source and destination nodes of each session randomly selected, and the minimum rate requirement is randomly selected within [1, 10].

We assume that the maximum transmission range at each node is 20. Correspondingly, the maximum transmission power at a node on a band m is $P^m_{\max} = (20)^{\alpha_m} P^m_T$, where the path loss index α_m is taken to be 4, and the transmission threshold P^m_T is set to $\eta W = 50\eta$. The maximum total transmission power at a node on all bands is set to $P_{\max} = 8P^m_{\max}$. We set the maximum interference range twice the maximum transmission range [21]. Then, the interference threshold is $P^m_I = (1/2)^{\alpha_m} P^m_T = (50/16)\eta$.

B. Case Study

Before we present complete simulation results, we first examine the iterative behavior and convergence of our distributed algorithm by simulation on one particular network instance. As shown in Fig. 8, this network instance has 50 nodes with five sessions (the source and the destination of each session are also marked in this figure). The minimum rate requirement of each session is 10.

Initially, all sessions start with zero scaling factor (see Fig. 9). Thus, the session with the smallest source node ID is chosen, e.g., session 1. A minimum-cost route, as well as scheduling and power control, for this session is constructed, and its scaling factor is increased to $K(1) = 8.73$ (corresponding to a session rate of 87.3). At the second iteration, there are four sessions with scaling factors of 0. Among these sessions, session 2 is chosen, and its scaling factor $K(2)$ is increased to 1.75. After five iterations, we have $K(1) = 8.73$, $K(2) = 1.75$, $K(3) = 15.92$, $K(4) = 17.24$, and $K(5) = 3.25$. Because $K(2) = 1.75$ is the smallest among the five iterations, we choose session 2 and try to increase its scaling factor in the next (sixth) iteration. During the sixth iteration, we find that it is infeasible to increase $K(2)$ under CIP. Thus, we resort to AIP, i.e., try to release some capacity from other sessions such that $K(2)$ may be increased. We find that $K(2)$ can be increased to 2.50 by decreasing $K(1)$ to 7.98 and $K(5)$ to 2.50. This iteration process continues. Finally, at the 14th iteration, we find that $K(2) = 5.24$ is the smallest scaling factor among the sessions. However, we find that it is not feasible to increase its scaling factor under either CIP or AIP. Thus, our algorithm terminates at this iteration, with $K = 5.24$.

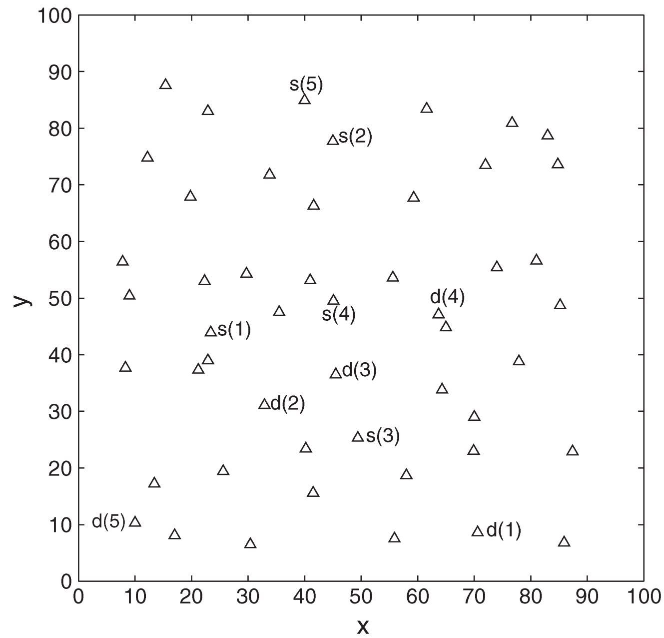

Fig. 8. Network topology for a 50-node CRN with five sessions.

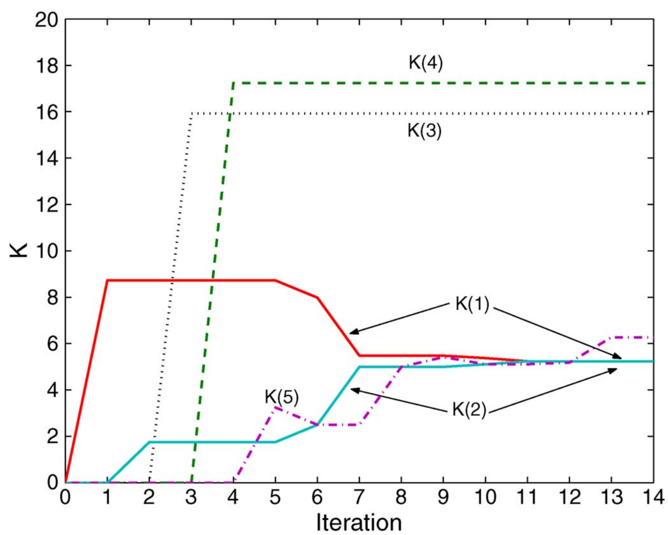

Fig. 9. Iterations of each session's scaling factor $K(l)$ in the simulation.

Note that our distributed algorithm offers a multipath routing solution as planned. In this network, session 5 uses two paths for flow routing in the final solution (see Fig. 10).

We now examine the performance of the distributed iteration. Using the upper bound developed in [20], we find that the upper bound for the scaling factor is 5.42. In our distributed algorithm, we have achieved a scaling factor of 5.24, which is 96.7% of

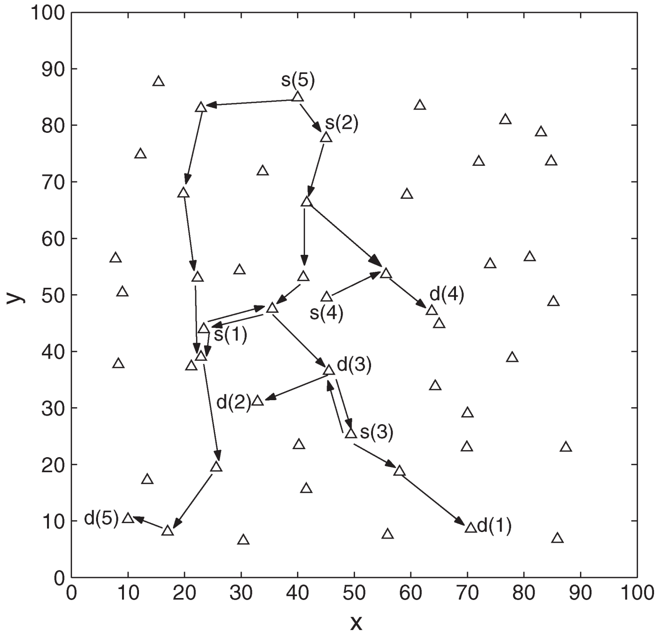

Fig. 10. Flow routing in the final solution for the 50-node CRN.

the upper bound. Although the maximum achievable (optimal) scaling factor is unknown, it is upper limited by this upper bound, that is, 5.42. Therefore, we conclude that the global scaling factor achieved by our distributed algorithm is *at least* 96.7% of the maximum.

In Section III-C, we derived an upper bound for the algorithm's complexity. In fact, our algorithm has a much lower complexity than that bound for most instances. As an example, for this 50-node network instance, the upper bound for the number of iterations is $2|\mathcal{N}||\mathcal{M}|(2|\mathcal{L}|^2+1)+2$, which is about $5.1 \cdot 10^4$. However, in the actual simulation, the algorithm terminates after 14 iterations. Our algorithm is targeted to long-term continuous traffic from each session (e.g., video). Because our algorithm can achieve optimal K reasonably fast, data transmission for a session can commence after its $K(l)$ converges.

C. Complete Simulation Results

We now present complete simulation results for all 100 network instances. For each network instance, we perform the same study as in Section IV-B. Fig. 11 shows the ratio of maximum achievable K obtained by our distributed algorithm normalized with respect to the upper bound for these 100 network instances. Note that, in some cases, the ratio is 100%, i.e., the result obtained by our distributed algorithm is identical to the respective upper bound obtained by relaxation. In such cases, the results found by our distributed algorithm are optimal. The average ratio for these 100 data is 83.5%, with a standard derivation of 0.13. Because the maximum achievable scaling factor (unknown) lies between the upper bound and the feasible solution obtained by our distributed algorithm, we conclude that the scaling factor obtained by our distributed algorithm must be even closer to the optimal solution.

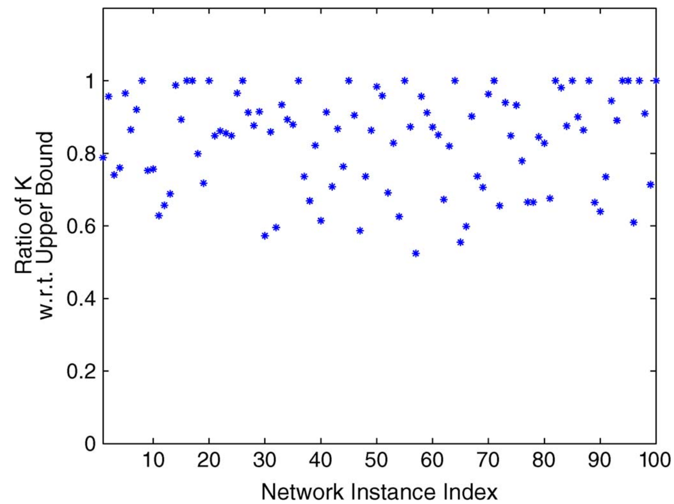

Fig. 11. Normalized scaling factor K for 100 network instances.

V. Related Work

Our related work review focuses on two lines of research that we find relevant: 1) multihop CRNs and 2) distributed optimization for wireless networks.

For multihop CRNs, there has recently been some active research [5], [22]–[24], [27], [28]. In [5], Chowdhury *et al.* proposed a transport protocol for multihop CRNs, which incorporated CR-related activities into feedback to the source. In [22], Shi and Hou studied joint power control, scheduling, and routing for optimal network resource allocation by a centralized approach. In [23], Steenstrup studied three different frequency assignment problems and designed distributed algorithms. In [24], Ugarte and McDonald found an upper bound on network capacity for multihop CRNs, although it is not clear how tight this bound is. In [27], Xin *et al.* studied how they can distributedly assign frequency bands at each node to form a topology such that a certain performance metric can be optimized. Their paper was based on the so-called fixed-channel approach, where the radio was assumed to operate on only one channel at a specific time. In [28], Zhao *et al.* proposed a distributed spectrum-sharing approach and showed that this approach offers throughput improvement over a dedicated channel approach. However, the performance gap between this approach and the optimal solution is unknown.

For distributed optimization for wireless networks, there has also been active research in recent years. Some of these efforts include distributed routing (e.g., [7], [12]), distributed scheduling (e.g., [1]–[4], [8], [17], [29]), and distributed power control (e.g., [18]). Furthermore, there have also been efforts that consider distributed optimization from a cross-layer perspective [6], [10], [11], [16]. In [6] and [11], cross-layer problems for joint optimization of routing and scheduling were studied. In these efforts, however, power control was not part of the optimization space. In [10], Lin and Shroff designed a distributed algorithm for maximizing the total session utility, which was shown to achieve a constant factor of the capacity region. Routing was not part of their optimization problem. In [16], Palomar and Chiang solved several maximizing network utility problems by some new distributed decomposition approaches. In their problems, scheduling was not considered, and the constraint on power control was relaxed to the sum of power on all links, instead of individual links.

VI. Conclusion

In this paper, we have investigated distributed optimization to maximize data rates for a set of user communication sessions in a multihop CRN. Our main contribution is the development of a distributed optimization algorithm that iteratively increases the data rate of the session with the minimum scaling factor (or the bottleneck session). Through simulation results, we compared the performance of the distributed optimization algorithm with an upper bound and showed that the performance of the proposed distributed algorithm is highly competitive.

References

[1] A. Brzezinski, G. Zussman, and E. Modiano, "Enabling distributed throughput maximization in wireless mesh networks—A partitioning approach," in *Proc. ACM MobiCom*, Los Angeles, CA, Sep. 24–29, 2006, pp. 26–37.

[2] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu, "Joint asynchronous congestion control and distributed scheduling for multihop wireless networks," in *Proc. IEEE INFOCOM*, Barcelona, Spain, Apr. 23–29, 2006, pp. 1–12.

[3] L. Chen, S. H. Low, and J. C. Doyle, "Joint congestion control and media access control design for ad hoc wireless networks," in *Proc. IEEE INFOCOM*, Miami, FL, Mar. 13–17, 2005, pp. 2212–2222.

[4] M. Chiang, S. Zhang, and P. Hande, "Distributed rate allocation for inelastic flows: Optimization frameworks, optimality conditions, and optimal algorithms," in *Proc. IEEE INFOCOM*, Miami, FL, Mar. 13–17, 2005, pp. 2679–2690.

[5] K. R. Chowdhury, M. Di Felice, and I. F. Akyildiz, "TP-CRAHN: A transport protocol for cognitive radio ad hoc networks," in *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, Apr. 19–25, 2009, pp. 2482–2490.

[6] A. Eryilmaz, A. Ozdaglar, and E. Modiano, "Polynomial complexity algorithms for full utilization of multihop wireless networks," in *Proc. IEEE INFOCOM*, Anchorage, AK, May 6–12, 2007, pp. 499–507.

[7] S. Guo, O. Yang, and V. Leung, "Approximation algorithms for longest lived directional multicast communications in WANETs," in *Proc. ACM MobiHoc*, Montreal, QC, Canada, Sep. 9–14, 2007, pp. 190–198.

[8] C. Joo, "A local greedy scheduling scheme with provable performance guarantee," in *Proc. ACM MobiHoc*, Hong Kong, May 26–30, 2008, pp. 111–120.

[9] Joint Tactical Radio System. [Online]. Available: http://www.globalsecurity.org/military/systems/ground/jtrs.htm

[10] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer rate control in wireless networks," in *Proc. IEEE INFOCOM*, Miami, FL, Mar. 13–17, 2005, pp. 1804–1814.

[11] X. Lin and S. Rasool, "A distributed joint channel assignment, scheduling and routing algorithm for multichannel ad hoc wireless networks," in *Proc. IEEE INFOCOM*, Anchorage, AK, May 6–12, 2007, pp. 1118–1126.

[12] Z. Li and B. Li, "Efficient and distributed computation of maximum multicast rates," in *Proc. IEEE INFOCOM*, Miami, FL, Mar. 13–17, 2005, pp. 1618–1628.

[13] X. Liu and W. Wang, "On the characteristics of spectrum-agile communication networks," in *Proc. IEEE DySPAN*, Baltimore, MD, Nov. 8–11, 2005, pp. 214–223.

[14] M. McHenry, "Spectrum white space measurements," presented at the New America Foundation BroadBand Forum, Jun. 20, 2003. [Online]. Available: http://www.newamerica.net/files/nafmigration/archive/Doc_File_185_1.pdf

[15] N. A. Lynch, *Distributed Algorithms*. San Francisco, CA: Morgan Kaufmann, 1996.

[16] D. P. Palomar and M. Chiang, "Alternative decompositions for distributed maximization of network utility: Framework and applications," in *Proc. IEEE INFOCOM*, Barcelona, Spain, Apr. 23–29, 2006, pp. 1–13.

[17] C. Peng, H. Zheng, and B. Y. Zhao, "Utilization and fairness in spectrum assignment for opportunistic spectrum access," *Mobile Netw. Appl.*, vol. 11, no. 4, pp. 555–576, Aug. 2006.

[18] S. S. Ram, V. V. Veeravalli, and A. Nedic, "Distributed nonautonomous power control through distributed convex optimization," in *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, Apr. 19–25, 2009, pp. 3001–3005.

[19] SAFECOM. [Online]. Available: http://www.safecomprogram.gov/SAFECOM/

[20] Y. Shi and Y. T. Hou, "A distributed optimization algorithm for multihop cognitive radio networks," in *Proc. IEEE INFOCOM*, Phoenix, AZ, Apr. 14–17, 2008, pp. 1966–1974.

[21] Y. Shi, Y. T. Hou, and S. Kompella, "How to correctly use the protocol interference model for multihop wireless networks," in *Proc. ACM MobiHoc*, New Orleans, LA, May 18–21, 2009, pp. 239–248.

[22] Y. Shi, Y. T. Hou, and H. Zhou, "Per-node-based optimal power control for multihop cognitive radio networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 10, pp. 5290–5299, Oct. 2009.

[23] M. E. Steenstrup, "Opportunistic use of radio-frequency spectrum: A network perspective," in *Proc. IEEE DySPAN*, Baltimore, MD, Nov. 8–11, 2005, pp. 638–641.

[24] D. Ugarte and A. B. McDonald, "On the capacity of dynamic spectrum-access-enabled networks," in *Proc. IEEE DySPAN*, Baltimore, MD, Nov. 8–11, 2005, pp. 630–633.

[25] Vanu Inc. [Online]. Available: http://www.vanu.com/solutions/solutions.html

[26] A. M. Wyglinski, M. Nekovee, and Y. T. Hou, Eds., *Cognitive Radio Communications and Networks: Principles and Practice*. Amsterdam, The Netherlands: Elsevier, 2010.

[27] C. Xin, B. Xie, and C.-C. Shen, "A novel layered-graph model for topology formation and routing in dynamic spectrum access networks," in *Proc. IEEE DySPAN*, Baltimore, MD, Nov. 8–11, 2005, pp. 308–317.

[28] J. Zhao, H. Zheng, and G. Yang, "Distributed coordination in dynamic spectrum allocation networks," in *Proc. IEEE DySPAN*, Baltimore, MD, Nov. 8–11, 2005, pp. 259–268.

[29] D. Zheng, W. Ge, and J. Zhang, "Distributed opportunistic scheduling for ad hoc communications: An optimal stopping approach," in *Proc. ACM MobiHoc*, Montreal, QC, Canada, Sep. 9–14, 2007, pp. 1–10.

Yi Shi (S'02–M'08) received the Ph.D. degree in computer engineering from Virginia Polytechnic Institute and State University (Virginia Tech), Blacksburg, in 2007.

He is currently a Research Scientist with the Bradley Department of Electrical and Computer Engineering, Virginia Tech. His research interests are focused on algorithms and optimization for cognitive radio networks, MIMO and cooperative communication networks, sensor networks, and ad hoc networks.

Dr. Shi was a recipient of the Best Paper Award at the IEEE Conference on Computer Communications (INFOCOM) in 2008 and the Chinese Government Award for Outstanding Ph.D. Students Abroad in 2006. He served as a Member of the Technical Program Committee for several major international conferences, including the 2009 ACM MobiHoc, and the 2009–2011 IEEE INFOCOM conferences.

Y. Thomas Hou (S'91–M'98–SM'04) received the Ph.D. degree in electrical engineering from the Polytechnic Institute of New York University, Brooklyn, in 1998.

From 1997 to 2002, he was a Researcher with the Fujitsu Laboratories of America, Sunnyvale, CA. Since 2002, he has been with the Bradley Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University (Virginia Tech), Blacksburg, where he is currently an Associate Professor. He is the holder of five U.S. patents. His research interests include cross-layer design and optimization for cognitive radio wireless networks, cooperative communications, MIMO-based ad hoc networks, video communications over dynamic ad hoc networks, and algorithm design for sensor networks.

Dr. Hou received the Young Investigator Award from the Office of Naval Research in 2003 and the National Science Foundation CAREER Award in 2004 for his research on optimizations and algorithm design for wireless ad hoc and sensor networks. He received five best paper awards from IEEE, including the 2008 IEEE INFOCOM Best Paper Award and the 2002 IEEE International Conference on Network Protocols (ICNP) Best Paper Award.

Huaibei (Barry) Zhou received the B.S. and M.S. degrees in physics from Wuhan University, Wuhan, China, in 1984 and 1987, respectively, and the Ph.D. degree in physics from the University of Maryland, College Park, in 1994.

From 1994 to 1996, he was a Postdoctoral Fellow with the National Institute of Standard and Technology, Gaithersburg, MD. From 1996 to 1999, he was a Senior Engineer with General Electric. From 1999 to 2002, he was a Senior Technical Manager with NEXTEL. He is currently the Dean of the International School of Software Engineering, Wuhan University. His research interests include cognitive radio and radio wave propagation.

Scott F. Midkiff (S'82–M'85–SM'92) received the B.S.E. and Ph.D. degrees in electrical engineering from Duke University, Durham, NC, and the M.S. degree in electrical engineering from Stanford University, Stanford, CA.

He was with Bell Laboratories and held a visiting position with Carnegie Mellon University, Pittsburgh, PA. In 1986, he joined Virginia Polytechnic Institute and State University (Virginia Tech), Blacksburg. From 2006 to 2009, he was a Program Director with the National Science Foundation. He is currently a Professor and the Department Head with the Bradley Department of Electrical and Computer Engineering, Virginia Tech. His research and teaching interests include wireless and ad hoc networks, network services for pervasive computing, and cyber-physical systems.