A Scheduling Algorithm for MIMO DoF Allocation in Multi-hop Networks

Huacheng Zeng, *Student Member, IEEE,* Yi Shi, *Senior Member, IEEE,* Y. Thomas Hou, *Fellow, IEEE,* Wenjing Lou, *Fellow, IEEE,* Hanif D. Sherali, Rongbo Zhu, *Member, IEEE,* Scott F. Midkiff, *Senior Member, IEEE*

Abstract—Recently, a new MIMO degree-of-freedom (DoF) model was proposed to allocate DoF resources for spatial multiplexing (SM) and interference cancellation (IC) in a multi-hop network. Although this DoF model promises many benefits, it hinges upon a *global* node ordering to keep track of IC responsibilities among all the nodes. An open question about this model is whether its global ordering property can be achieved among the nodes in the network through distributed operations. In this paper, we explore this question by studying DoF scheduling in a multi-hop MIMO network, with the objective of maximizing the minimum throughput among a set of sessions. We propose an efficient DoF scheduling algorithm to solve it and show that our algorithm only requires local operations. We prove that the resulting DoF scheduling solution is globally feasible and show that there exists a corresponding feasible global node ordering for IC, albeit such global ordering is implicit. Simulation results show that the solution values obtained by our algorithm are relatively close to the upper bound values computed by CPLEX solver, thereby indicating that our algorithm is highly competitive.

Index Terms—Scheduling, multi-hop wireless networks, MIMO, degree-of-freedom (DoF), throughput maximization.

1 Introduction

In recent years, MIMO has attracted a growing interest in the wireless networking research community due to its ability to offer significant increases in data throughput without additional bandwidth or transmit power [26]. Among the research efforts of MIMO in multi-hop networks, there is an active research line that builds upon the so-called degree-of-freedom (DoF) model [1], [2], [4], [8], [12], [15], [16], [17], [24], [25]. The concept of DoF was originally defined to represent the maximum multiplexing gain of MIMO channel in the information theory (IT) community [11], [31]. It was then extended by the networking research community to characterize a node's spatial freedom provided by its multiple antennas. Typically, the number of available DoFs at a node is assumed to be equal to the number of antennas at the node and represents the total available resources at the node for *spatial multiplexing* (SM) and *interference cancellation* (IC) [3], [7], [13], [23]. SM refers to the use of one or multiple DoFs for data stream transmission/reception (at both transmit and receive nodes), with each data stream corresponding to one DoF. IC refers to the use of one or multiple DoFs to cancel interference, which can be done either at the transmit node or the receive node. For example, consider the two links in Fig. 1. To transmit z_1 data streams on link (T_1, R_1), both nodes T_1 and R_1 need to consume z_1 DoFs for SM. Similarly, to transmit z_2 data streams on link (T_2, R_2), both nodes T_2 and R_2 need to consume z_2 DoFs for SM. The interference from T_2 to R_1 can be canceled by either R_1 or T_2. If R_1 cancels this interference, it needs to consume z_2 DoFs. If T_2 cancels this interference, it needs to consume z_1 DoFs.

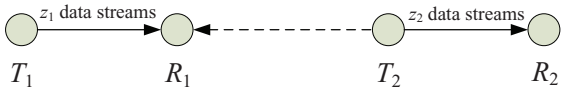


Fig. 1. An example that illustrates SM and IC.

A significant advantage of DoF model is that it only requires simple numeric computation (addition and subtraction) to keep track of SM and IC at a node. Although a DoF model is not able to completely capture all the PHY-layer capabilities of MIMO, it offers a simple yet effective tool to study MIMO in a multi-hop network. As such, various DoF models have been proposed and applied to solve a variety of network problems (see, e.g., [1], [4], [8], [12] for throughput maximization, [2], [15], [17], [24], [25] for MAC protocols).

Since interference can be canceled by either its transmit node or its receive node (as shown in Fig. 1), a question to ask is which node should take the responsibility for IC? The lack of a systematic rule in assigning IC responsibility is likely to lead to either sub-optimal or infeasible solution and is the main limitation in the prior efforts. In [15], Mumey *et al.* proposed an approximation algorithm for joint stream control and scheduling, where IC is done only at the receiver. In [24], Sundaresan *et al.* studied MAC design in which IC can be done only at the receiver. Without exploiting IC at the transmit nodes, the DoF model in [15], [24] tends to shrink the feasible

- *H. Zeng, Y. Shi, Y.T. Hou, W. Lou, H.D. Sherali, and S.F. Midkiff are with Virginia Tech, Blacksburg, VA, 24061 USA. E-mail: {zeng, yshi, thou, wjlou, hanifs, midkiff}@vt.edu. (Corresponding author: Y.T. Hou)*
- *R. Zhu is with South-Central University for Nationalities, China. E-mail: rongbozhu@gmail.com.*

Manuscript received August 25, 2014; revised January 19, 2015; accepted March 5, 2015.

solution space unnecessarily. In [25], Sundaresan *et al.* proposed to allocate only one DoF at each node for SM while reserving the remaining DoFs for IC. Such a static approach cannot be optimal for maximizing throughput in MIMO networks. Park *et al.* [17] studied a MAC problem and proposed to impose the IC responsibility to a newly active link (without changing the current DoF behavior at other links). Again, such non-collaborative IC strategy is overly restrictive and is unlikely to lead to an optimal solution. In [1], [8], the authors studied cross-layer design for throughput maximization problems. Although both efforts allowed either transmit or receive node to perform IC, there was no clear guideline on how this should be done in a systematic manner. As a result, it was shown in [21] that such an approach results in a small DoF region that is far from optimal. In [2], Blough *et al.* proposed a scheduling algorithm for throughput maximization based on a DoF model, which allowed IC to be done at either transmitter or receiver without any other restriction. However, due to the lack of mathematical proof, their DoF allocation cannot guarantee a feasible solution at the PHY layer.

In [21], we explored the important problem of how to use DoF correctly (to ensure feasibility) and efficiently (to avoid duplication in IC). The main result in [21] is a new DoF model that performs IC among the nodes based on a *global node ordering*. Specifically, each transmit/receive node only consumes DoFs for canceling interference to/from those nodes before itself in the global node ordering, it does not need to consume DoFs for canceling interference to/from those nodes after itself in the global node ordering. It was shown in [21] that once DoF allocation is performed at each node following a global node ordering, potential duplication in IC can be completely eliminated. Furthermore, such a DoF allocation is guaranteed to be feasible at the PHY layer. We will review more details of this global node ordering concept in Section 2.

Despite its performance superiority over previous MIMO DoF models, the new DoF model in [21] relies on a *global* node ordering to keep track of IC responsibilities among all the nodes. A natural question that one may raise about this model is whether such characteristics would have difficulty in a network where all operations are performed distributedly. In this paper, we explore this question by applying this model to a multi-hop MIMO network. We are interested in whether carefully designed local node operations (and local node ordering) can be translated into the desired ordering and feasibility on the global level. Specifically, for a set of sessions in the network, we study how to schedule DoF resources among the nodes so that the minimum data throughput among the sessions can be maximized.[1] We formulate this throughput maximization problem as a cross-layer optimization problem and develop an efficient and fast DoF scheduling algorithm to solve it. Our DoF scheduling algorithm is an iterative algorithm and includes three modules in each iteration: *link selection module* (LSM), *resource allocation module* (RAM), and *local re-adjustment module* (LRM). Some highlights of our algorithm include:

- It is amenable to local implementation. We show that each module in our algorithm can be implemented in a distributed manner.
- Upon algorithm termination, the final DoF allocation solution is feasible at global level. That is, there exists a global node ordering for the final solution corresponding to the DoF allocation at each node. This is not trivial, given that each module performs operations locally without global knowledge.
- Its performance is highly competitive. Simulation results show that the objective values obtained by our algorithm are close to upper bounds of the same problem (obtained by CPLEX solver). We therefore conclude that the objective by our algorithm is very close to the optimum.
- It has a polynomial-time complexity and thus offers a solution rather quickly (in contrast to exponential complexity of solving the MILP problem).

The remainder of this paper is organized as follows. In Section 2, we give a review of the new DoF model in [21]. In Section 3, we formulate a network throughput maximization problem based on the new DoF link model. Section 4 states our problem and Section 5 presents our DoF scheduling algorithm. Section 6 proves the global feasibility of final solution and Section 7 analyzes the algorithm. Section 8 presents our simulation results and Section 9 concludes this paper.

2 Background: A New MIMO DoF Model

We consider a multi-hop network consisting of a set of nodes, each of which is equipped with multiple antennas. Assume that the channel matrix between any two nodes has full rank. Then the number of DoFs available to a node is equal to the number of its antennas. A node can use some or all of its DoFs for either SM or IC, as long as the number of consumed DoFs does not exceed the total available DoFs at the node. For the MIMO DoF model in [21], DoFs (for SM and IC) at a node is allocated based on the following guideline.

For SM, both the transmit and receive nodes consume DoFs. The number of DoFs consumed at both the transmit and receive nodes is equal to the number of data streams that is transported between the two nodes. For IC, unlike SM, only a transmit node or a receive node needs to consume DoFs, not both. The questions of which node should be responsible for IC and how many DoFs are needed are effectively addressed by the node ordering concept described in [21]. Specifically, all nodes in the network are put into an ordered list. The position of a node in the list represents its order in the node list. A node consumes DoFs for IC as follows:

1. Note that our DoF scheduling problem differs from those efforts on distributed MIMO scheduling (e.g., [14], [18]) as the latter was not based on a DoF link model.

- *Transmit node.* If the node is a transmit node, then it only needs to cancel its interference to those receive nodes (within its interference range) that are before itself in the ordered node list. It does not need to consume DoFs to cancel its interference to those receive nodes that are after itself in the ordered node list. Interference from this transmit node to those receive nodes after itself will be canceled by those receive nodes latter. For IC, the number of DoFs consumed at this transmit node is equal to the total number of data streams received by those receive nodes from their intended transmitters.
- *Receive node.* If the node is a receive node, then it only needs to cancel interference from those transmit nodes (whose interference ranges cover this receive node) that are before itself in the ordered node list. It does not need to cancel interference from those transmit nodes that are after itself in the ordered node list. Interference from those transmit nodes after this node will be canceled by those transmit nodes latter. For IC, the number of DoFs consumed at this receive node is equal to the total number of data streams transmitted by those transmit nodes.

As shown in [21], by referencing an ordered node list, one can avoid duplication in IC between transmit and receive nodes while ensuring the feasibility of the final DoF scheduling solution. Furthermore, an optimal ordering of a node list can be obtained by putting the ordering constraint into a problem formulation. In the rest of this section, we give a model for the ordering-based DoF allocation in a time-slotted system. Table 3 (in supplemental material) lists the notation in this paper.

Assume that a time frame in data plane (time resource for data transmission and reception) consists of T equal-length time slots. Suppose that there are N nodes in the network and node i has A_i antennas. Denote a binary variable $x_i(t)$ as an indicator of whether node i is a transmitter in time slot t. Similarly, denote $y_i(t)$ as an indicator of whether node i is a receiver in time slot t. Let $\mathcal{L}_i^{\text{in}}$ and $\mathcal{L}_i^{\text{out}}$ be the set of possible incoming and outgoing links at node i (determined by the transmission range of a node), respectively. Denote $z_l(t)$ as the number of data streams on link l in time slot t. Then we have

$$x_i(t) \le \sum_{l\in\mathcal{L}_i^{\text{out}}} z_l(t) \le A_i \cdot x_i(t), \quad (1 \le i \le N, 1 \le t \le T), \tag{1}$$

$$y_i(t) \le \sum_{l\in\mathcal{L}_i^{\text{in}}} z_l(t) \le A_i \cdot y_i(t), \quad (1 \le i \le N, 1 \le t \le T). \tag{2}$$

Denote $\pi(t)$ as the order of nodes in the network in time slot t and denote $\pi_i(t)$ as the position of node i in order $\pi(t)$. Then we have

$$1 \le \pi_i(t) \le N, \quad (1 \le i \le N, 1 \le t \le T). \tag{3}$$

Denote binary variable $\theta_{ji}(t)$ as the relative position of nodes j and i in order $\pi(t)$ as follows: $\theta_{ji}(t) = 1$ if node j is before node i in order $\pi(t)$ and 0 otherwise. Then we have

$$\pi_i(t) - N \cdot \theta_{ji}(t) + 1 \le \pi_j(t) \le \pi_i(t) - N \cdot \theta_{ji}(t) + N - 1, \quad (1 \le i \le N, j \in \mathcal{I}_i, 1 \le t \le T), \tag{4}$$

where $\mathcal{I}_i$ is the set of nodes within node i's interference range.

For each node i in the ordered node list, we can mathematically model its DoF consumption for SM and IC as follows:

$$\text{If } x_i(t) = 1, \text{ then } \sum_{l\in\mathcal{L}_i^{\text{out}}} z_l(t) + \sum_{j\in\mathcal{I}_i} \theta_{ji}(t) \sum_{k\in\mathcal{L}_j^{\text{in}}}^{\text{Tx}(k)\neq i} z_k(t) \le A_i, \quad (1 \le i \le N, 1 \le t \le T), \tag{5}$$

where on the left-hand side of the inequality, the first and second terms represent the number of DoFs consumed by node i for SM and IC, respectively. Similarly,

$$\text{If } y_i(t) = 1, \text{ then } \sum_{l\in\mathcal{L}_i^{\text{in}}} z_l(t) + \sum_{j\in\mathcal{I}_i} \theta_{ji}(t) \sum_{k\in\mathcal{L}_j^{\text{out}}}^{\text{Rx}(k)\neq i} z_k(t) \le A_i, \quad (1 \le i \le N, 1 \le t \le T). \tag{6}$$

For constraint (5), if $x_i(t) = 1$, then we have $\sum_{l\in\mathcal{L}_i^{\text{out}}} z_l(t) + \sum_{j\in\mathcal{I}_i} \theta_{ji}(t) \sum_{k\in\mathcal{L}_j^{\text{in}}}^{\text{Tx}(k)\neq i} z_k(t) \le A_i$. On the other hand, if $x_i(t) = 0$, then no DoF is consumed. Constraint (5) can be reformulated by incorporating binary variable $x_i(t)$ into the expression as follows:

$$\sum_{l\in\mathcal{L}_i^{\text{out}}} z_l(t) + \sum_{j\in\mathcal{I}_i} \theta_{ji}(t) \sum_{k\in\mathcal{L}_j^{\text{in}}}^{\text{Tx}(k)\neq i} z_k(t) \le A_i x_i(t) + (1 - x_i(t))B, \quad (1 \le i \le N, 1 \le t \le T), \tag{7}$$

where $B = \sum_{i=1}^{N} A_i$ is an upper bound of the second term on the left-hand side of (7).

Similarly, constraint (6) can be reformulated as follows:

$$\sum_{l\in\mathcal{L}_i^{\text{in}}} z_l(t) + \sum_{j\in\mathcal{I}_i} \theta_{ji}(t) \sum_{k\in\mathcal{L}_j^{\text{out}}}^{\text{Rx}(k)\neq i} z_k(t) \le A_i y_i(t) + (1 - y_i(t))B, \quad (1 \le i \le N, 1 \le t \le T). \tag{8}$$

3 From DoF Link Model to Network-level Throughput Maximization

The above DoF link model offers an excellent tool to study networking problems for MIMO networks. Consider a MIMO network consisting of a set of N nodes, where node i, $i = 1, 2, \cdots, N$, is equipped with A_i antennas. Suppose that there is a set of F unicast sessions in the network, with their source and destination nodes being randomly selected among all the nodes. The route of each session can be computed by some routing protocol (e.g., AODV and OLSR). We assume that scheduling is done in a time frame consisting of T time slots. In such a network, our goal is to find an

optimal scheduling solution in each time slot so that the minimum achievable (end-to-end) throughput among all the sessions can be maximized. Denote $r(f)$ as the achievable end-to-end throughput of session f. Then our objective can be mathematically written as: maximize $\min_{1\le f\le F}\{r(f)\}$.

3.1 Formulation

Half Duplex. We assume that a node's transceiver is half-duplex. Then we have

$$x_i(t)+y_i(t)\le 1, \quad (1\le i\le N; 1\le t\le T). \tag{9}$$

Link Capacity Constraint. Denote $\mathrm{src}(f)$ and $\mathrm{dst}(f)$ as the source and destination nodes of session f, respectively. Denote $r_l(f)$ as the amount of data rate on link l that is attributed to session f. For simplicity, we assume that fixed modulation and coding scheme (MCS) is used for each data stream and that each data stream corresponds to one unit data rate. Then the average rate of link l over T time slots is $\frac{1}{T}\sum_t z_l(t)$. Thus, we have

$$\sum_{f=1}^{F} r_l(f)\le \frac{1}{T}\sum_{t=1}^{T} z_l(t), \quad (1\le l\le L), \tag{10}$$

where L is the number of links in the network.

Flow Balance at Each Node. At each node, flow conservation must be observed. Then at a source node, we have

$$\sum_{l\in\mathcal{L}_i^{\mathrm{out}}} r_l(f)=r(f), \quad (i=\mathrm{src}(f), 1\le f\le F). \tag{11}$$

At an intermediate node, we have

$$\sum_{l\in\mathcal{L}_i^{\mathrm{out}}} r_l(f)=\sum_{l\in\mathcal{L}_i^{\mathrm{in}}} r_l(f), \quad (1\le i\le N, 1\le f\le F, i\ne \mathrm{src}(f), i\ne \mathrm{dst}(f)). \tag{12}$$

At a destination node, we have

$$\sum_{l\in\mathcal{L}_i^{\mathrm{in}}} r_l(f)=r(f), \quad (i=\mathrm{dst}(f), 1\le f\le F). \tag{13}$$

It can be easily verified that if (11) and (12) are satisfied, then (13) is also satisfied. Therefore, it is sufficient to include only (11) and (12) in the problem formulation.

Throughput Objective. Denote $r_{\min}$ as the throughput rate of the bottleneck session. Then we have

$$r(f)\ge r_{\min}, \quad (1\le f\le F). \tag{14}$$

Based on the above constraints and the MIMO DoF model described in Section 2, our throughput optimization problem can be formulated in Fig. 2.

OPT-DoF-Raw:
Max $r_{\min}$
S.t. MIMO DoF constraints: (1)–(4) and (7)–(8);
half-duplex constraints: (9);
link capacity constraints: (10);
flow balance constraints: (11) and (12);
throughput objective: (14).

Fig. 2. A formulation for the DoF scheduling problem.

OPT-DoF:
max $r_{\min}$
s.t. MIMO DoF constraints: (1)–(4) and (15)–(20);
half-duplex constraints: (9);
link capacity constraints: (10);
flow balance constraints: (11) and (12);
throughput objective: (14).

Fig. 3. A reformulation for the DoF scheduling problem.

3.2 Reformulation-Linearization

The formulation in Fig. 2 is in the form of mixed integer nonlinear program (MINLP). It is possible to reformulate the nonlinear constraints into linear ones. The nonlinear constraints in Fig. 2 are (7) and (8). To linearize them, we employ the *reformulation-linearization technique* (RLT) [20], which replaces nonlinear terms by introducing new variables and new linear constraints. We define $\lambda_{ji}(t)=\theta_{ji}(t)\sum_{k\in\mathcal{L}_j^{\mathrm{in}}}^{\mathrm{Tx}(k)\ne i} z_k(t)$. Then we can replace the nonlinear constraint (7) by the following linear constraints:

$$\sum_{l\in\mathcal{L}_i^{\mathrm{out}}} z_l(t)+\sum_{j\in\mathcal{I}_i}\lambda_{ji}(t)\le A_i x_i(t)+(1-x_i(t))B, \quad (1\le i\le N, 1\le t\le T), \tag{15}$$

$$0\le \lambda_{ji}(t)\le A_j\cdot\theta_{ji}(t), \quad (1\le i\le N, j\in\mathcal{I}_i, 1\le t\le T), \tag{16}$$

$$A_j\cdot\theta_{ji}(t)-A_j+\sum_{k\in\mathcal{L}_j^{\mathrm{in}}}^{\mathrm{Tx}(k)\ne i} z_k(t)\le\lambda_{ji}(t)\le\sum_{k\in\mathcal{L}_j^{\mathrm{in}}}^{\mathrm{Tx}(k)\ne i} z_k(t), \quad (1\le i\le N, j\in\mathcal{I}_i, 1\le t\le T). \tag{17}$$

Similarly, we define $\mu_{ji}(t)=\theta_{ji}(t)\sum_{k\in\mathcal{L}_j^{\mathrm{out}}}^{\mathrm{Rx}(k)\ne i} z_k(t)$. Then we can replace the nonlinear constraint (8) by the following linear constraints:

$$\sum_{l\in\mathcal{L}_i^{\mathrm{in}}} z_l(t)+\sum_{j\in\mathcal{I}_i}\mu_{ji}(t)\le A_i y_i(t)+(1-y_i(t))B, \quad (1\le i\le N, 1\le t\le T), \tag{18}$$

$$0\le \mu_{ji}(t)\le A_j\cdot\theta_{ji}(t), \quad (1\le i\le N, j\in\mathcal{I}_i, 1\le t\le T), \tag{19}$$

$$A_j\cdot\theta_{ji}(t)-A_j+\sum_{k\in\mathcal{L}_j^{\mathrm{out}}}^{\mathrm{Rx}(k)\ne i} z_k(t)\le\mu_{ji}(t)\le\sum_{k\in\mathcal{L}_j^{\mathrm{out}}}^{\mathrm{Rx}(k)\ne i} z_k(t), \quad (1\le i\le N, j\in\mathcal{I}_i, 1\le t\le T). \tag{20}$$

Based on the linearized constraints, the formulation in Fig. 2 can be reformulated in Fig. 3, which is in the form of MILP. It is not difficult to see that the problem

formulation inevitably requires integer and binary variables. This indicates that the problem we are trying to solve is NP-hard in general [5], [19], although a formal proof is not given in this paper.

4 Problem Statement

Goals. Although there already exist algorithms (e.g., Branch-and-Bound and sequential fixing [9]) and commercial optimization solvers (e.g., IBM CPLEX [10]) that can be used to solve the problem in Fig. 3, these algorithms/solvers are limited to the centralized environment. The goal of this paper is to develop a distributed and efficient algorithm to solve the problem in Fig. 3. Meanwhile, we hope that the resulting DoF scheduling solution (found by our algorithm) will be globally feasible in each time slot. By "globally feasible", we mean that at the PHY layer, there exist a set of precoding vectors at each transmitter and a set of decoding vectors at each receiver so that the data streams on each link can be transported free of interference using zero-forcing technique. For readers who are interested in understanding why the node ordering-based DoF model is feasible at the PHY layer, we refer them to [21], [29]. In short, instead of dealing with complex design of precoding and decoding vectors at the PHY layer, the node ordering concept in [21], [29] allows us to ensure the global feasibility of the DoF scheduling solution at the PHY layer.

Challenges. To make sure that the resulting final DoF scheduling solution is global feasible at the PHY layer, we must make sure there exists a global node ordering in each time slot in the network. As explained in Section 2, the relative ordering between two nodes directly determines DoF consumption responsibility at each node for IC. In a centralized environment, an optimal global node ordering can be found by putting the ordering constraints (3) and (4) into the problem formulation (see Fig. 3). However, in a distributed multi-hop network environment, each node can only exchange scheduling information with its neighboring nodes to establish and maintain some local relative ordering among neighboring nodes. It is not clear how such a distributed local ordering in each individual node can lead to a feasible global ordering among all the nodes in the network.

In our distributed algorithm, through proper design, we show that it is possible to have a per-node based local node ordering match to a global ordering of all nodes in the network, thereby achieving the same effect as that in a centralized environment. Specifically, we will show that the establishment of initial per-node based local node ordering and re-adjustment of neighboring node ordering during each iteration lead to a feasible global node ordering.

5 A Distributed Algorithm

In this section, we develop a distributed DoF allocation algorithm to solve the problem in Fig. 3. We first state our assumptions and then give an overview of the algorithm. Finally, we explain the key modules of the algorithm in detail.

5.1 Assumptions

We have the following assumptions in the design of our distributed algorithm.

- Network and Traffic: We assume that the network is static (with little node mobility). We also assume that the nodes in the network are synchronized at a resolution to the time slot level, which is not stringent [22]. We also assume that each session has a persistent and latency-tolerant traffic at its source. When the network topology or traffic pattern is changed, the scheduling algorithm should run again so that the network resource can be reallocated accordingly.
- Channel State Information (CSI): A node is assumed to have CSI between itself and its neighboring nodes. The CSI can be obtained as follows. A node periodically broadcasts a public pilot sequence such that the CSI between them can be estimated. During the data communication, the estimated CSI can be used as CSIR (CSI at receiver-side) for SM and IC. Based on the reciprocity property of wireless channel, the estimated CSI can also be used as CSIT (CSI at transmitter-side) for SM and IC. For a static network, although wireless channel is still time-variant, its changing speed is slow and the channel has a long coherent time. This allows us to obtain relatively accurate CSI with infrequent updates. In such an environment, the overhead in CSI acquisition tends to be acceptable and has been investigated in [27], [30], which showed that it is practical to obtain CSI for MIMO's SM and IC in a stationary network with acceptable overhead.
- Control Channel: We assume there is a control channel for scheduling (e.g., IEEE 802.16j mobile multi-hop relay (MMR) networks [6]) and the control channel consists of a set of time slots. In each time slot, a node may need to exchange scheduling information with the nodes within its interference range. A question to ask is if a node can communicate with those nodes within its interference range (beyond its transmission range) since the data communication is limited in the transmission range. In wireless network (e.g., cellular network), the transmit power of control plane is usually larger than that of data plane, allowing control plane to have a larger coverage than data plane. Given that the scheduling information exchange is within the control plane, it is practical for a node to exchange scheduling information with the nodes within its interference range.
- Paper Scope: The goal of this paper is to outline an efficient algorithm that can solve the problem in Fig. 3 in a distributed environment. The main

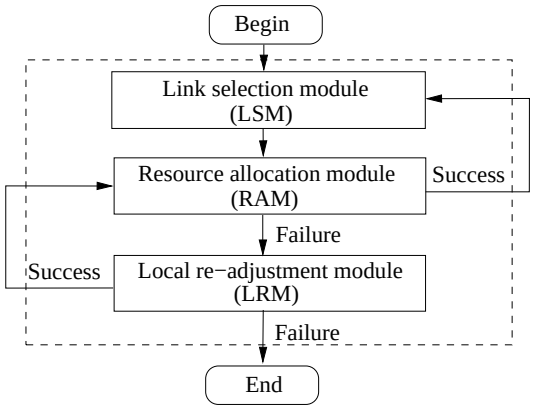


Fig. 4. A flow chart of our DoF scheduling algorithm.

contribution is to show the proposed algorithm can preserve global feasibility in DoF allocation even through local operations. From protocol and implementation perspective, we recognize that there remain many details that need to be spelled out. Due to space limitation, we defer such details for future work and instead focus our efforts on algorithm design in this paper.

5.2 Algorithm Overview

We offer an overview of the proposed DoF allocation algorithm to solve the optimization problem in Fig. 3. A flow chart of the algorithm is illustrated in Fig. 4, which includes three key modules: *link selection module* (LSM), *resource allocation module* (RAM), and *local re-adjustment module* (LRM). In essence, it is an iterative greedy algorithm that attempts to increase the minimum rate among all links in each iteration. After a bottleneck link is identified, the RAM is invoked to see how the DoFs for SM can be increased in one of the T time slots while satisfying all local and neighboring interference constraints. If RAM is not able to yield a feasible increment, then we explore whether altering the local ordering of some nodes may yield a feasible increment, despite that a global node ordering information is not available to each individual node. This is done by LRM.

Here we give an overview of each module.

- **LSM.** The goal of this module is to identify a link for rate increment in an iteration. We propose a *session-independent* link selection approach by establishing a list of all links in the network based on their potential "interference burden". We show that this link selection approach is equivalent to the session-dependent link selection approach in terms of increasing the minimum rate among all sessions. We also show that this link selection approach can be implemented in a distributed environment.
- **RAM.** The goal of this module is to allocate DoF resource in one of the T time slots to increase the rate of the selected link. We first introduce local node ordering and global node ordering as well as the data structure that should be maintained at each node. Then, we explore the conditions under which the rate of the selected link can be increased in a given time slot and how DoFs should be allocated for the rate increment if the conditions are satisfied. We further show that if the DoF allocation *before* the rate increment is feasible based on a global node ordering, then the DoF allocation *after* the rate increment is also feasible and corresponds to a new global node ordering. Based on the outcome for the rate increment in a given time slot, we explain how to allocate DoF resource for the rate increment in a time frame.
- **LRM.** When RAM fails to increase the rate of the selected link, it is likely that the DoF allocation algorithm is stuck in a local optimal point. To allow the algorithm to jump out the local optimal point, we use LRM to alter some local node ordering so that some DoFs can be relieved from some nodes to accommodate one more data stream on the selected link. In a given time slot, we first identify the set $\mathcal{D}$ of nodes that are in shortage of DoF resource for the rate increment, and then explain how to adjust the local ordering for a node in $\mathcal{D}$ so that its remaining DoFs can be increased. We show that such local node ordering adjustment can preserve global feasibility of a solution and the existence of a global node ordering, albeit implicit. Based on the outcome of the local ordering adjustment in a given time slot, we explain how to perform the local ordering adjustment in a time frame.

In the rest of this section, we explain the three modules in more detail.

5.3 Link Selection Module

Several approaches may be considered to increase the minimum rate among all sessions iteratively. A straightforward approach is to identify a session with the minimum rate in the network and then try to increase the rate of each link by one unit along the session's path. Unfortunately, our simulation results reveal that an algorithm based on this approach does not perform well. The failure of such a *session-dependent* link selection approach may be attributed to the fact that it ignores the significance of potential "interference burden" of each link in the network. By "interference burden" of a link, we mean the number of DoFs required at both the link's transmitter and receiver for IC. This consideration motivates us to pursue a *session-independent* link selection approach based on a link's interference burden in the network.

More formally, for node i, we define its interference burden as q_i, which is the number of nodes within node i's interference range. Then for link (i, j), we define its interference burden (or priority) as $q_{(i,j)} = q_i + q_j$. In our approach, we sort all the links in the network based on *non-increasing* order of their interference burden into a

list, which we denote as $\mathcal{B}$. A small but important detail in $\mathcal{B}$ is the representation of a link that is traversed by multiple sessions. In our design, we would like to have session-independent link based approach to achieve the same effect as the session-dependent link based approach in term of increasing the minimum rate among all sessions iteratively. To do this, it is necessary to represent a link multiple times in list $\mathcal{B}$ if it is traversed by multiple sessions. An example is given in supplemental material to illustrate how to establish list $\mathcal{B}$.

Based on this link list $\mathcal{B}$, we select link sequentially for rate increment (by one data stream). The reason why we consider links with higher priorities (i.e., larger interference burden) first is that resource allocation task for these links is likely to be more demanding than those links with lower priorities (i.e., smaller interference burden). Intuitively, once these most demanding links are taken care of first, it would be easier for us to perform resource allocation for those less demanding links with the remaining network resource.

Since list $\mathcal{B}$ is invisible to each link in a distributed network, the question to ask is how each link can obtain its rank in list $\mathcal{B}$. This problem can be solved by using the distributed ranking algorithm in [28]. To apply the distributed ranking algorithm to our problem, we can have the transmitter of each link maintain the priority of that link and then execute the distributed ranking algorithm by treating the reciprocal priority of that link (i.e., $1/q_{(i,j)}$) as its initial value. At the end of the ranking algorithm, the transmitter of each link can obtain the rank of that link. Given that the ranking algorithm in [28] has two phases and the node operations in each phase do not require synchronization, it takes two time slots in control channel for the transmitter of each link to obtain its rank. In the worst case, the communication overhead of the ranking algorithm requires $N^2/2+O(N)$ messages, which are acceptable in practical networks.

Once each active link obtains its rank in list $\mathcal{B}$, then one link in $\mathcal{B}$ will be selected in each time slot (in control channel) to schedule rate increment. Such link selection process is cyclic as time slots in control channel progress. As a result, for each link, it has precise knowledge of which time slots it will be chosen for rate increment operations.

5.4 Resource Allocation Module

The goal of the RAM is to allocate DoF resource for the selected link so that the rate of the selected link can be increased by one data stream in one of the T time slots on the data plane. To do this, we first discuss the relationship between local node ordering and global node ordering. Based on this understanding, we introduce the data structure that should be maintained at each node, and then explore the condition under which a link rate can be successfully increased by one data stream.

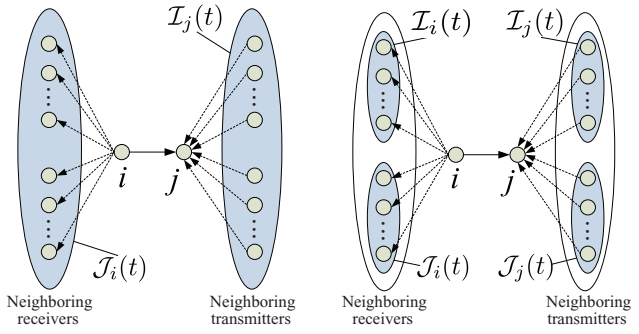
Local Node Ordering vs. Global Node Ordering. Recall that in Section 2, a global node ordering plays a key role in a feasible and efficient DoF scheduling [21]. However, in a distributed environment it is impractical to establish and maintain such a global node ordering in the network. Instead, we propose to have each node establish and maintain a relative ordering with its neighboring nodes in a distributed environment. In Section 6, we will show that the established and maintained local ordering at each individual node leads to a feasible global node ordering.

To establish a local ordering, we have each node i maintain two sets of its neighboring nodes: (i) $\mathcal{I}_i(t)$ the set of nodes for which node i has allocated DoFs for IC: these nodes are considered *before* node i in the local ordering; and (ii) $\mathcal{J}_i(t)$ the set of nodes that have allocated their DoFs to cancel interference either from or to node i: these nodes are considered *after* node i in the local ordering. We will show how these two sets can be established and maintained through a distributed mechanism. More importantly, we will show that by properly updating and maintaining these two sets at each node, one can determine which node is responsible for the cancellation of a particular interference. Further, we show in Theorem 1 that based on these two sets, one can identify a corresponding global node ordering in the network, although none of the nodes has such an explicit knowledge.

TABLE 1
State information at each node i.

Symbol	Definition
$s_i(t)$	The status of node i (transmit, receive, or idle) in time slot t
$\mathcal{I}_i$	The set of nodes in node i's interference range
$\mathcal{I}_i^T(t)$	Transmitters in $\mathcal{I}_i$ in time slot t
$\mathcal{I}_i^R(t)$	Receivers in $\mathcal{I}_i$ in time slot t
$\mathcal{L}_i$	The set of incoming and outgoing links at node i
$\{z_l(t): l \in \mathcal{L}_i\}$	The number of data streams on the incoming or outgoing links of node i
$\lambda_i^{\text{SM}}(t)$	The number of DoFs at node i allocated for SM in time slot t
$\lambda_i^{\text{IC}}(t)$	The number of DoFs at node i allocated for IC in time slot t
$\mathcal{T}_i(t)$	The set of nodes to which node i has established links in time slot t
$\mathcal{I}_i(t)$	The set of nodes for which node i has allocated DoFs for IC in time slot t
$\mathcal{J}_i(t)$	The set of node i's neighboring nodes that have allocated their DoFs to cancel interference either to or from nodes i in time slot t

Data Structure at a Node. Table 1 lists state information that we maintain at each node in the network. In the table, $\mathcal{I}_i(t)$ and $\mathcal{J}_i(t)$ are the two sets representing the local ordering at node i. Since $\mathcal{I}_i(t)$ contains the set of nodes for which node i has allocated DoFs for IC in time slot t, we consider that these nodes are *before* node i in the local ordering. Similarly, since $\mathcal{J}_i(t)$ contains the set of node i's neighboring nodes that have allocated their DoFs to cancel interference either to or from nodes i in time slot t, these nodes are *after* node i in the local ordering. $\lambda_i^{\text{SM}}(t)$ is the number of DoFs allocated for SM at node i in time slot t. $\lambda_i^{\text{IC}}(t)$ is the number of DoFs allocated for IC at node i in time slot t. For ease of explanation, denote



(a) Case I: link (i, j) is not active (b) Case II: link (i, j) is active

Fig. 5. An example that illustrates RAM.

$\overline{\lambda}_i(t)$ as the number of remaining DoFs at node i in time slot t. Thus, we have $\overline{\lambda}_i(t) = A_i - \lambda_i^{\text{SM}}(t) - \lambda_i^{\text{IC}}(t)$. $s_i(t)$ is the status of node i in time slot t, which is defined as follows: $s_i(t) =$ "T" if node i is a transmitter; $s_i(t) =$ "R" if node i is a receiver; and $s_i(t) =$ "I" if node i is idle.

During the initialization stage, each node is set to idle status, i.e., $s_i(t) =$ "I" for $1 \le i \le N$, $1 \le t \le T$; each node allocates zero DoF for SM and IC, i.e., $\lambda_i^{\text{SM}}(t) = 0$ and $\lambda_i^{\text{IC}}(t) = 0$, $z_l(t) = 0$, $\mathcal{T}_i(t) = \emptyset$, $\mathcal{I}_i(t) = \emptyset$, $\mathcal{J}_i(t) = \emptyset$ for $1 \le i \le N$, $1 \le l \le L$, $1 \le t \le T$.

Rate Increment in Time Slot t. To increase the rate of link (i, j) by one data stream in time slot t, nodes i and j first check their current status in time slot t. If the status for both nodes meet the requirements (in Case I or Case II below), then nodes i and j as well as their neighboring nodes will check whether they have enough remaining DoFs for IC under their current local orderings. If yes, nodes i and j and relevant neighboring nodes update their state information to accommodate this increment on link (i, j).

Case I: Referring to Fig. 5(a), link (i, j) is not active and we wish to add one data stream on this link if the following conditions are satisfied: (i) node i is idle; (ii) all of the receivers within node i's interference range have at least one remaining DoF to cancel interference from node i; (iii) node j is idle and its total DoFs are more than the sum of DoFs for SM at those nodes that are interfering node j (assuming node j will become the last node in the local ordering).

If the above conditions are satisfied, then nodes i and j as well as their neighboring nodes do the following:

- At node i, its status is changed from idle to transmit. The number of DoFs consumed for SM at node i is updated to one. The rate of link (i, j) is increased to one. To update the local ordering at node i, we define node i to be the first node in its local ordering. To do this, we update $\mathcal{J}_i(t) = \mathcal{I}_i^R(t)$.
- At each of node i's neighboring nodes $a \in \mathcal{J}_i(t)$, node a adds node i into set $\mathcal{I}_a(t)$ and increases its DoF consumption (for IC) by one.
- At node j, its status is updated from idle to receive. The number of DoFs consumed for SM is updated to one. The rate of link (i, j) is updated to one. Correspondingly, to update the local ordering at node j, we define node j to be the last node in its local ordering. To do this, we update $\mathcal{I}_j(t) = \mathcal{I}_j^T(t)$. The number of DoFs consumed for IC is updated to be the sum of data streams of its neighboring transmitters except node i.
- At each of node j's neighboring nodes $b \in \mathcal{I}_j(t)$, node b adds node j into its set $\mathcal{J}_b(t)$.

Case II: Referring to Fig. 5(b), link (i, j) is already active and we wish to add one more data stream on this link if the following conditions are satisfied: (i) node i is a transmitter and has at least one DoF remaining for SM; (ii) each node in $\mathcal{J}_i(t)$ has at least one DoF remaining to cancel interference from node i; (iii) node j is a receiver and has at least one DoF remaining for SM; (iv) each node in $\mathcal{J}_j(t)$ has at least one DoF remaining to cancel its interference to node j.

If the above conditions are satisfied, then nodes i and j as well as their neighboring nodes do the following:

- At node i, the number of DoFs consumed for SM is increased by one. The rate of link (i, j) is increased by one.
- Each node in $\mathcal{J}_i(t)$ increases its DoF consumption for IC by one.
- At node j, the number of DoFs consumed for SM is increased by one. The rate of link (i, j) is increased by one.
- Each node in $\mathcal{J}_j(t)$ increases its DoF consumption for IC by one.

A pseudo-code of rate increment in a given time slot is given in Fig. 11 (in supplemental material). It is easy to see that rate increment (as described in Cases I and II) is a local operation and also feasible (in terms of DoF allocation) for those nodes involved in this operation. A natural question to ask is how such local operation will affect global feasibility among all nodes. We now state an important property, which says that if the DoF allocation is feasible among all the nodes in the network, then this local operation will result in a new DoF allocation that is also globally feasible.

Formally, denote $\pi(t)$ as a global ordering for all nodes in the network. Based on $\pi(t)$, suppose that $\varphi(t)$ is a feasible DoF scheduling for SM and IC at all nodes in the network. Denote $\hat{\varphi}(t)$ as the new DoF scheduling after the rate increment operation on $\varphi(t)$. Then we have the following lemma:

Lemma 1: $\hat{\varphi}(t)$ is a globally feasible DoF scheduling. Further, there exists a global ordering $\hat{\pi}(t)$ that corresponds to $\hat{\varphi}(t)$.

Proof. We show that the lemma holds for two cases.

For Case I, we construct the global ordering $\hat{\pi}(t)$ by letting $\hat{\pi}(t) = [i\ \pi(t)\ j]$. Then we check each node's DoF consumption (for SM and IC) in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$ as follows: (i) Node i only needs to consume one DoF for SM and does not need to consume DoF for IC. Thus node i has enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$. (ii) For each node in $\mathcal{I}_i^R(t)$, since it has at least

one remaining DoF, it can perform SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$. (iii) Given $\sum_{h\in\mathcal{I}_j^T(t)}\lambda_h^{\text{SM}}(t) < A_j$, node j has enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$. (iv) For all other nodes in $\hat{\pi}(t)$, since they have enough DoFs for SM and IC in $\varphi(t)$ based on $\pi(t)$, they also have enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$. Therefore, $\hat{\varphi}(t)$ is a globally feasible DoF scheduling and $\hat{\pi}(t)$ is a global ordering that corresponds to $\hat{\varphi}(t)$.

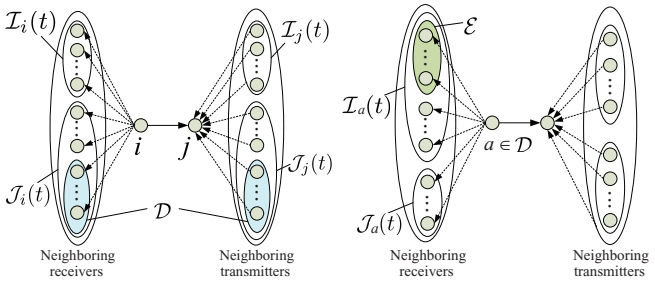
For Case II, we construct the global ordering $\hat{\pi}(t)$ by letting $\hat{\pi}(t) = \pi(t)$. Now we check each node's DoF consumption (for SM and IC) in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$: (i) Node i has at least one remaining DoF. Since node i only needs one more DoF for SM, it has enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$. (ii) Every node in $\mathcal{J}_i(t)$ has at least one remaining DoF. Since each of these nodes only needs one more DoF for IC, all nodes in $\mathcal{J}_i(t)$ have enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$. (iii) Node j has at least one remaining DoF. Since node j only needs one more DoF for SM, it has enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$. (iv) Every node in $\mathcal{J}_j(t)$ has at least one remaining DoF. Since each of these nodes only needs one more DoF for IC, all these nodes in $\mathcal{J}_j(t)$ have enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$. (v) For all other nodes in $\hat{\pi}(t)$, since they have enough DoFs for SM and IC in $\varphi(t)$ based on $\pi(t)$, they have enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$. Therefore, $\hat{\varphi}(t)$ is a globally feasible DoF scheduling and $\hat{\pi}(t)$ is a global ordering that corresponds to $\hat{\varphi}(t)$. □

Resource Allocation in a Time Frame. Recall that there are T time slots in a time frame. If the rate increment operation described above fails in the first time slot, we try it again in the second time slot and so forth, until it is successful in a time slot or fails after all T time slots. It is not difficult to see that this module is amenable to local implementation as all operations of this module are restricted on the selected link and its neighboring links.

5.5 Local Re-adjustment Module

When RAM fails to increase one data stream on the chosen link, it is likely that the algorithm is stuck in a local optimal point. Following the flow chart in Fig. 4, we invoke the *local re-adjustment module* (LRM) to allow the algorithm to jump out of the local optimal point. The goal of this module is to adjust the local ordering for the nodes associated with the chosen link so that IC responsibilities can be transferred from one node to another, thereby relieving some DoF resources for some nodes so as to accommodate a new data stream on the chosen link.

Local Ordering Adjustment in Time Slot t. Given that RAM fails to increase a data stream on a given link (i, j), we conclude that there is a lack of DoF resources at a subset of nodes among i, j, $\mathcal{J}_i$, or $\mathcal{J}_j$ based on the current local ordering at these nodes. This subset of nodes can be easily identified in a hypothesized scenario by looking for those nodes that would use more DoFs than their total DoFs if one more data stream were added on link (i, j). Denote $\mathcal{D}$ as this subset of nodes in shortage of DoF resources, as shown in Fig. 6(a).



(a) Set $\mathcal{D}$ for link (i, j) (b) Set $\mathcal{E}$ for node $a \in \mathcal{D}$

Fig. 6. An example that illustrates LRM.

For each node $a \in \mathcal{D}$, we perform local ordering adjustment, with the goal of relieving one DoF (already used for IC) from a node so that a new DoF can become available. To avoid race condition in a distributed system, we use a token and let it pass from one node to the next in $\mathcal{D}$ so that at any time, only one node is allowed to perform local ordering adjustment. The token can be firstly initiated and held by node i, and then passed on to node j. The token is passed to the next node in $\mathcal{D}$ only if the local ordering adjustment in the previous node in $\mathcal{D}$ is successful (resulting in one free DoF at that node). Otherwise, the token will not be passed to the next node and we move on to the next time slot in a frame.

Referring to Fig. 6(b), for a given node $a \in \mathcal{D}$ that currently holds the token, we first need to identify a set of a's neighboring nodes $\mathcal{E}$ that can relieve some of a's DoF consumption for IC. First, nodes in $\mathcal{E}$ should not include any node of i, j, and their neighboring nodes. Otherwise, we may run into a loop of changing local node ordering without yielding any net improvement. Second, nodes in $\mathcal{E}$ must be ahead of a in a's local ordering, i.e., $b \in \mathcal{I}_a(t)$, since a is using its DoFs to cancel interference from nodes in $\mathcal{E}$. Third, nodes in $\mathcal{E}$ should have enough remaining DoF resources to relieve a's DoF consumption for IC to b, i.e., $\overline{\lambda}_b(t) \geq \lambda_a^{\text{SM}}(t)$. Finally, we need to ensure that there does not exist another node, say c, that is in a higher local order than node $b \in \mathcal{E}$ (i.e., c is after b) but in a lower local order than a (i.e., c is before a). This will ensure that a local node ordering swap between a and b will not violate the local ordering between b and c.

For the set of candidate nodes in $\mathcal{E}$ for node a, we only need one node to swap its local ordering with a. In our algorithm, we choose a node in $\mathcal{E}$ that has the most number of remaining DoFs. A tie can be broken by selecting the node with a smaller node ID. Denote this node in $\mathcal{E}$ as b^*. For nodes a and b^*, we perform the following operation: (i) node a moves b^* from its $\mathcal{I}_a(t)$ to $\mathcal{J}_a(t)$, indicating that node b^* is now behind node a in the new ordering; (ii) node a no longer needs to cancel interference from node b^* and its remaining DoFs are increased, i.e. $\lambda_a^{\text{IC}}(t) := \lambda_a^{\text{IC}}(t) - \lambda_{b^*}^{\text{SM}}(t)$, $\overline{\lambda}_a(t) := \overline{\lambda}_a(t) +$

$\lambda_{b^*}^{\text{SM}}(t)$; (iii) node b^* moves a from its $\mathcal{J}_{b^*}(t)$ to $\mathcal{I}_{b^*}(t)$, indicating that node a is now before node b^* in the new ordering; (iv) node b^* now needs to cancel interference from node a and its remaining DoFs are decreased, i.e., $\lambda_{b^*}^{\text{IC}}(t) := \lambda_{b^*}^{\text{IC}}(t) + \lambda_a^{\text{SM}}(t)$, $\overline{\lambda}_{b^*}(t) := \overline{\lambda}_{b^*}(t) - \lambda_a^{\text{SM}}(t)$. A pseudo-code of local ordering adjustment is given in Fig. 12 (in supplemental material).

A question to ask is how such a local node reordering operation will affect global feasibility among all the nodes. We now state an important property, which says that if the DoF scheduling is feasible among all the nodes in the network, then the LRM operation will result in a new DoF scheduling that is also globally feasible. Formally, denote $\pi(t)$ as a global ordering for all the nodes in the network. Based on $\pi(t)$, suppose $\varphi(t)$ is a feasible DoF scheduling for SM and IC for all nodes in the network. Denote $\hat{\varphi}(t)$ as the DoF scheduling for all the nodes after LRM is performed at some nodes a and b^* under $\varphi(t)$. Then we have the following lemma:

Lemma 2: $\hat{\varphi}(t)$ is a globally feasible DoF scheduling. Further, there exists a global ordering $\hat{\pi}(t)$ that corresponds to $\hat{\varphi}(t)$.

Proof. The proof is based on construction. Denote B as the set of nodes before b^* in $\pi(t)$, C as the set of nodes after a in $\pi(t)$, D as the set of nodes between a and b^* in $\pi(t)$. Then, we have $\pi(t) = [B\ b^*\ D\ a\ C]$. Denote Γ as the set of nodes that are in a lower ordering than node a. We construct the global ordering $\hat{\pi}(t)$ by letting $\hat{\pi}(t) = [B\ \ D\cap\Gamma\ \ a\ \ b^*\ \ D\cap\Gamma^c\ \ C]$, where Γ^c is the complement set of Γ. Next, we check the DoF consumption for SM and IC at each node in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$.

- For each node $h \in B\cup C$: From $\pi(t)$ to $\hat{\pi}(t)$, the local ordering of node h does not change. Since node h has enough DoFs for SM and IC in $\varphi(t)$ based on $\pi(t)$, it also has enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$.
- For each node $h \in D \cap \Gamma$: From $\pi(t)$ to $\hat{\pi}(t)$, the nodes in $\{b^*\}\cup(D\cap\Gamma^c)$ are moved from the positions before node h to the positions after node h. Thus, the DoF consumption at node h in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$ is less than or equal to that in $\varphi(t)$ based on $\pi(t)$. Therefore, node h has enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$.
- For node a: From $\pi(t)$ to $\hat{\pi}(t)$, the local ordering of node a does not change except that node b^* is moved from $\mathcal{I}_a(t)$ to $\mathcal{J}_a(t)$. Thus, node a does not need to cancel the interference from/to node b^* in $\hat{\pi}(t)$, indicating that the DoF consumption of node a in $\hat{\varphi}(t)$ is less than that in $\varphi(t)$. Therefore, node a has enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$. Further, the number of remaining DoFs at node a is increased by $\lambda_{b^*}^{\text{SM}}(t)$.
- For node b^*: Since there does not exist any node in a higher local order than node b^* and in a lower local order than node a, we know that from $\pi(t)$ to $\hat{\pi}(t)$, the local ordering of node b^* does not change except that node a is moved from $\mathcal{J}_{b^*}(t)$ to $\mathcal{I}_{b^*}(t)$. Thus, node b^* needs to cancel the interference from/to node a in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$. Since $\overline{\lambda}_{b^*}(t) \geq \lambda_a^{\text{SM}}(t)$, there are enough remaining DoFs at node b^* to cancel the interference from/to node a. Therefore, node b^* has enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$.
- For each node $h \in D\cap\Gamma^c$: From $\pi(t)$ to $\hat{\pi}(t)$, the local ordering of node h does not change, indicating that the DoF consumption of node h in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$ is the same as that in $\varphi(t)$ based on $\pi(t)$. Therefore, node h also has enough DoFs for SM and IC in $\hat{\varphi}(t)$ based on $\hat{\pi}(t)$.

Since every node in $\hat{\varphi}(t)$ has enough DoFs for SM and IC based on $\hat{\pi}(t)$, we conclude that $\hat{\varphi}(t)$ is a globally feasible DoF scheduling and $\hat{\pi}(t)$ is a global ordering that corresponds to $\hat{\varphi}(t)$. □

Local Ordering Adjustment in a Time Frame. Recall that there are T time slots in a time frame. If the local ordering adjustment described above fails in the first time slot, we try again in the second time slot and so forth, until local ordering adjustment is successful or fails after all T time slots. It is easy to see that this module is amenable to local implementation as all operations of this module are restricted on the selected link and its neighboring links.

6 Proving Global Feasibility of Final Solution

Recall that both RAM and LRM in our algorithm perform local operations and are amenable to distributed implementation. A natural question to ask is whether the final DoF scheduling at all nodes in the network (upon the algorithm termination) is still feasible at a global level. The following theorem answers this question.

Theorem 1: Suppose that $\varphi(t)$ is the final DoF scheduling for SM and IC at all nodes in the network. Then, $\varphi(t)$ is a globally feasible solution. Further, there exists a global ordering $\pi(t)$ that corresponds to $\varphi(t)$.

Proof. We prove it by induction. Denote $\varphi_n(t)$ as the DoF scheduling at all nodes in the network at the end of the n-th iteration. It is easy to see that the iterative algorithm in Fig. 4 will terminate in a finite number of iterations.

Base case: For $n = 1$, we show that $\varphi_n(t)$ is a global feasible solution with a global ordering $\pi_n(t)$. To see this, note that before the first iteration, none of the DoFs at any node in the network is allocated. So the LSM selects the link with the highest priority, say link (i, j). We perform RAM for the selected link (i, j) and thus obtain DoF scheduling $\varphi_1(t)$. Since there are no other active nodes in the network, $\varphi_1(t)$ is also a global feasible solution. Further, there exists a trivial global node ordering $\pi_1(t) = [i\ j]$ that corresponds to $\varphi_1(t)$.

Inductive step: Suppose that $\varphi_n(t)$ is a global feasible solution with a global ordering $\pi_n(t)$. We show that at the end of the $(n+1)$-th iteration, $\varphi_{n+1}(t)$ is also a global feasible solution with a global ordering $\pi_{n+1}(t)$. From $\varphi_n(t)$ to $\varphi_{n+1}(t)$, the operations may include RAM only

or both LRM and RAM (see Fig. 4). From Lemma 2, we know that LRM will preserve global feasibility of a solution as well as the existence of a global node ordering. From Lemma 1, we know that RAM will also preserve the global feasibility of a solution as well as the existence of a global node ordering. Therefore, if $\varphi_n(t)$ is a feasible solution with a global ordering $\pi_n(t)$, then $\varphi_{n+1}(t)$ is a feasible solution with a global ordering $\pi_{n+1}(t)$.

Combining the base case and the inductive step, we have that the final DoF scheduling $\varphi(t)$ is a globally feasible solution. Further, there exists a global ordering $\pi(t)$ that corresponds to $\varphi(t)$. □

7 Algorithm Analysis

7.1 Piecing up Global Node Ordering

In Section 6, we showed that the final DoF scheduling $\varphi(t)$ is a globally feasible solution and there exists a global ordering $\pi(t)$ that corresponds to $\varphi(t)$. Since the global ordering $\pi(t)$ is invisible to each individual node in the network, one may wonder how to piece up the global ordering $\pi(t)$ based on all the local node orderings in the network.

Denote $\mathcal{S}$ as the set of active nodes in time slot t in the final DoF scheduling solution. To obtain the global ordering $\pi(t)$, we first initialize $\pi(t)$ to an empty node list and then iteratively insert a node from $\mathcal{S}$ to $\pi(t)$ as follows:

Step 1: From $\mathcal{S}$, select a node i with $\mathcal{I}_i(t) = \emptyset$.
Step 2: Add node i at the end of node list $\pi(t)$, i.e., $\pi(t) := [\pi(t) \quad i]$.
Step 3: Remove node i from $\mathcal{S}$ and remove node i from $\mathcal{I}_j(t)$ for each $j \in \mathcal{S}$.
Step 4: If $\mathcal{S}$ is not empty, then go to Step 1.

A pseudo-code for finding a global node ordering $\pi(t)$ is given in Fig. 13 (in supplemental material). Note that there may exist multiple global node orderings that correspond to the final DoF scheduling in a given time slot.

7.2 Computational Complexity

We now show that the DoF scheduling algorithm in Fig. 4 is of polynomial time complexity. For LSM, its computation mainly consists of sorting the links along the path of sessions in the network. As we explained in Section 5.3, the LSM can be done in a distributed fashion by employing the distributed ranking algorithm in [28], which has $O(N^4)$ complexity.

For RAM, nodes i and j (transmitter and receiver of the selected link) need to check the feasibility of increasing one data stream in T time slots. In each time slot, the computation mainly consists of two parts: (i) nodes i and j as well as their neighboring nodes check their remaining DoFs, which has $O(N)$ complexity; (ii) nodes i and j as well as their neighboring nodes update their state information, which has $O(N)$ complexity. Since these operations can be done in T time slots, the complexity of RAM is $O(NT)$. For LRM, nodes i and j as well as their neighboring nodes perform local ordering adjustment in T time slots. In each time slot, the computation mainly consists of three parts: (i) identifying the subset of nodes $\mathcal{D}$, which has complexity $O(N)$; (ii) identifying a set of nodes $\mathcal{E}$ for each node $a \in \mathcal{D}$, which has complexity $O(N^2)$; (iii) adjusting the local ordering for each node $a \in \mathcal{D}$, which has complexity $O(N^2)$. Since these operations can be done in T time slots, the complexity of LRM is $O(N^2T)$. Following the flow chart in Fig. 4, the computation of each iteration mainly consists of RAM and LRM. Therefore, the complexity of each iteration of the DoF scheduling algorithm is $O(N^2T)$.

For this DoF scheduling algorithm, the number of iterations is bounded by $O(N^2TA)$. Since the complexity of each iteration is $O(N^2T)$, the total computational complexity is $O(N^4T^2A)$.

7.3 Overhead Analysis

In a distributed environment, the success of this DoF scheduling algorithm relies on the message exchange in control channel between neighboring nodes. Since it is hard to precisely quantify overhead induced by this algorithm, we develop an upper bound for the total volume of message exchanges between the neighboring nodes in this algorithm. In what follows, we analyze the volume of message exchanges in each module (LSM, RAM, and LRM) in the worst case.

For LSM, message exchanges are required to sort the links in the network. As we explained in Section 5.3, LSM sorts the links by directly employing the distributed ranking algorithm in [28]. Based on the results in [28], LSM requires $O(N^2)$ message exchanges in the worst case. For RAM and LRM, they are iterative modules and there are at most $O(N^2TA)$ iterations. To characterize their total volume of message exchanges, we first analyze their volume of message exchanges in each iteration. For RAM, nodes i and j need to check T time slots and the volume of message exchanges in each time slot is $O(1)$. So RAM requires $O(T)$ message exchanges in an iteration. For LRM, nodes i and j need to check T time slots and the volume of message exchanges in each time slot is $O(N)$. So LRM requires $O(NT)$ message exchanges in an iteration. Since there are at most $O(N^2TA)$ iterations, the total volume of message exchanges induced by RAM and LRM is $O(N^3T^2A)$. By adding the message exchanges induced by the three modules together, this DoF scheduling algorithm requires $O(N^3T^2A)$ message exchanges to achieve convergence in the worst case (when there is no interference). But in a practical network when there exists interference, the algorithm will converge much faster than $O(N^2TA)$ and thus will incur much less message overhead.

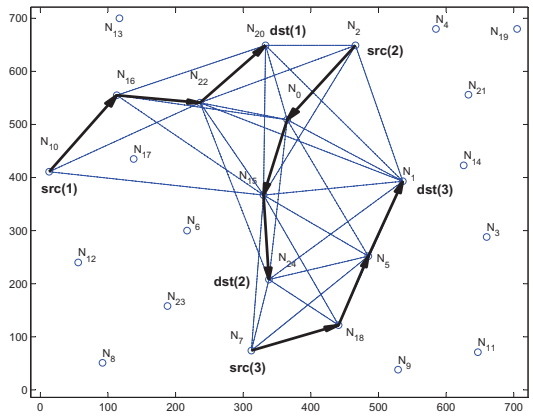


Fig. 7. A 25-node network instance.

TABLE 2
Local ordering and DoF consumption at each node in the first time slot.

Node i	$\mathcal{I}_i(t)$	$\mathcal{J}_i(t)$	$\lambda_i^{\text{SM}}(t)$	$\lambda_i^{\text{IC}}(t)$
N_0	$\{N_{22}\}$	$\{N_{15}, N_5\}$	2	1
N_1	$\{N_{15}, N_{22}\}$	$\{N_2\}$	2	2
N_2	$\{N_1\}$	$\{N_{20}\}$	2	2
N_5	$\{N_0\}$	$\{N_{18}, N_{24}\}$	2	2
N_7	$\{N_{24}\}$	$\emptyset$	1	1
N_{10}	$\emptyset$	$\emptyset$	2	0
N_{15}	$\{N_0\}$	$\{N_{18}, N_1, N_{16}, N_{20}\}$	1	2
N_{16}	$\{N_{15}, N_{22}\}$	$\emptyset$	2	2
N_{18}	$\{N_{15}, N_5\}$	$\emptyset$	1	3
N_{20}	$\{N_2, N_{15}\}$	$\emptyset$	1	3
N_{22}	$\emptyset$	$\{N_0, N_1, N_{16}, N_{24}\}$	1	0
N_{24}	$\{N_5, N_{22}\}$	$\{N_7\}$	1	3

8 Performance Evaluation

In this section, we present simulation results to demonstrate the performance of the proposed DoF scheduling algorithm. Ideally, the best performance benchmark would be an optimal solution to the OPT-DoF problem in Fig. 3. However, OPT-DoF formulation is an MILP and its optimal solution cannot always be obtained (even with a large amount of time) by a solver such as CPLEX [10]. But one thing we can do is to compare the *performance* of the proposed algorithm against an upper bound of OPT-DoF, which can be obtained by solving the problem formulation in Fig. 3 by CPLEX with a fixed termination time (e.g., 3 hours).[2] Note that the optimal solution value lies between the upper bound and the feasible solution value found by our algorithm. Therefore, if simulation results show that the objective value found by our algorithm is close to the upper bound computed by CPLEX, then we can infer with confidence that the result by our algorithm is even closer to the optimal solution (thus highly competitive).

In addition to a comparison between our algorithm and an upper bound from CPLEX, we also compare our algorithm with a simple DoF allocation algorithm where each node allocates DoF simply based on the current state of its neighbors without performing local ordering adjustment. Such a simple DoF allocation algorithm is the same as the RAM module in our algorithm but without the more complex LRM module. Therefore, the objective value from the simple DoF allocation algorithm can be obtained through running the RAM module in our algorithm (without LRM).

8.1 Simulation Setting

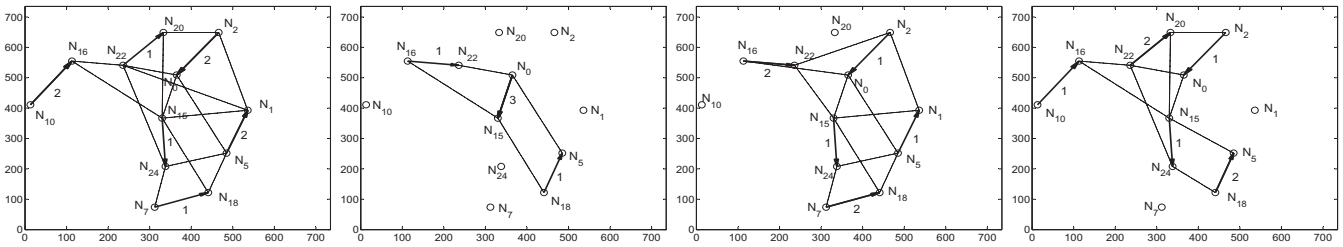
For ease of exposition, we normalize all units for distance, time, bandwidth, and data rate with appropriate dimensions. We consider networks of three sizes: (i) 25 nodes in a 750×750 area with 3 sessions; (ii) 50 nodes in a 1000×1000 area with 4 sessions; and (iii) 100 nodes in a 1500×1500 area with 5 sessions. We assume that all transmit nodes have the same transmission range 180 and the same interference range 360. For each network size, 100 randomly generated network instances are studied. For each network instance, the source and destination nodes of each session are randomly selected, with the route between them being shortest path route. We assume that each node is equipped with four antennas and there are four time slots in a time frame.

2. When using CPLEX to solve MILP problems, it always yields a lower bound (a feasible solution) and an upper bound for the objective value. The gap between the lower and the upper bounds becomes smaller as time continues. If the lower bound coincides with the upper bound, then an optimal solution is found.

8.2 A Case Study

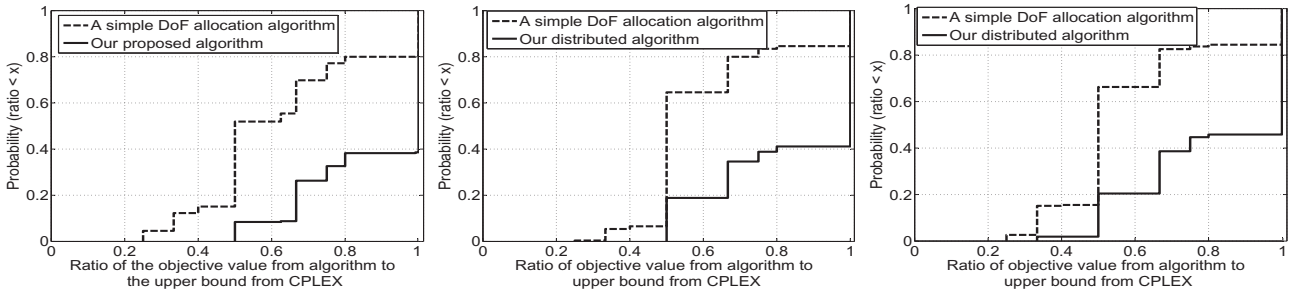
In this subsection, we study a 25-node network instance as shown in Fig. 7. In this figure, the solid arrow line represents link while the dashed line represents potential interference. There are three sessions in this network as shown in the figure. Figure 8 gives the details of the solution found by our algorithm in four time slots. Let's consider the first time slot as an example. The set of active links are (N_{10}, N_{16}), (N_{22}, N_{20}), (N_2, N_0), (N_{15}, N_{24}), (N_7, N_{18}), and (N_5, N_1). The two local node sets for each active node in this time slot are given in Table 2. For example, for node N_0, it considers N_{22} before itself (i.e., $\mathcal{I}_{N_0}(1) = \{N_{22}\}$) while N_{15} and N_5 after itself (i.e., $\mathcal{J}_{N_0}(1) = \{N_{15}, N_5\}$). Node N_0 uses two DoFs for SM to receive two data streams from N_2 (i.e., $\lambda_{N_0}^{\text{SM}}(1) = 2$) and uses one DoF to cancel interference from N_{22} (i.e., $\lambda_{N_0}^{\text{IC}}(1) = 1$). Also for this time slot, we can find a global node ordering for the DoF scheduling solution based on each node's local ordering, which is $[N_{10}, N_{22}, N_0, N_5, N_{15}, N_{16}, N_{18}, N_{24}, N_1, N_2, N_7, N_{20}]$. It is easy to verify that for this time slot, each node has enough DoFs for SM and IC based on this global ordering, indicating that the final DoF scheduling solution is globally feasible.

The objective value found by our algorithm is 0.75, corresponding to 3 data streams in 4 time slots for a bottleneck session. The upper bound by CPLEX is also 0.75, which shows that our solution is optimal in this case study. The objective value found by the simple DoF allocation algorithm is 0.5, which means our algorithm



(a) Time slot 1. (b) Time slot 2. (c) Time slot 3. (d) Time slot 4.

Fig. 8. Active links in each time slot in the 25-node case study.



(a) 25-node network (b) 50-node network (c) 100-node network

Fig. 9. The CDF of the ratio of the objective value from two algorithms to the upper bound from CPLEX.

has 50% throughput improvement compared to this simple DoF allocation algorithm.

8.3 Complete Simulation Results

We now present complete simulation results for three network sizes (25, 50, and 100 nodes). For each network size, we generate 200 random network instances. Figure 9 present the CDFs of the ratio of the objective value found by algorithm to the upper bound obtained by CPLEX for each network size. In each figure, there are two curves: the solid one is for our proposed algorithm and the dashed one is for the simple DoF allocation algorithm. We see that both CDF curves are not smooth but follow staircase shape. This is because the feasible objective value of the problem in Fig. 3 is discrete.

Based on the CDFs in Fig. 9, we can calculate the average results for each network size. The average ratio of objective value found by our proposed algorithm over CPLEX upper bound can achieve 84.1% for 25-node network, 83.2% for 50-node network, and 81.5% for 100-node network, respectively. Since the optimal solution lies between the feasible solution by our algorithm and the upper bound by CPLEX solver, we conclude that our solution is very close to the optimum. Moreover, the ratio of average objective value found by our proposed algorithm has 35.2% improvement over the simple DoF scheduling algorithm for 25-node network, 38.3% improvement for 50-node network, and 40.3% improvement for 100-node network, respectively.

9 Conclusions

A recent advance in MIMO link model allows research of MIMO on a network scale (i.e., multi-hop MIMO networks). However, such new MIMO link model hinges upon a global node ordering to keep track of IC responsibilities among the nodes, which is centralized by default. The goal of this paper is to show that it is possible to develop a distributed MIMO scheduling algorithm that achieves both global IC feasibility and an implicit global node ordering even if all operations are performed at a local level among neighboring nodes. The proposed MIMO DoF scheduling algorithm has both a greedy component as well as an aggressive component to counter potential trap of a local optimum. Simulation results show that it is able to achieve objective values very close to upper bound results by CPLEX, which is a very stringent benchmark to measure competitiveness.

References

[1] R. Bhatia and L. Li, "Throughput optimization of wireless mesh networks with MIMO links," in *Proc. IEEE INFOCOM*, pp. 2326–2330, Anchorage, AK, May 2007.

[2] D. Blough, G. Resta, P. Santi, R. Srinivasan, and L.M. Cortes-Pena, "Optimal one-shot scheduling for MIMO networks," in *Proc. IEEE SECON*, pp. 377–385, Salt Lake City, Utah, June 2011.

[3] L.-U. Choi and R.D. Murch, "A transmit preprocessing technique for multiuser MIMO systems using a decomposition approach," *IEEE Trans. on Wireless Communications*, vol. 3, no. 1, pp. 20–24, Jan. 2004.

[4] C. Gao, Y. Shi, Y.T. Hou, and S. Kompella, "On the throughput of MIMO-empowered multi-hop cognitive radio networks," *IEEE Trans. on Mobile Computing*, vol. 10, no. 11, pp. 1505–1519, Nov. 2011.

[5] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Chapter 1–2, W.H. Freeman and Company, New York, 1979.

[6] V. Genc, S. Murphy, Y. Yang, and J. Murphy, "IEEE 802.16J relay-based wireless access networks: an overview," *IEEE Wireless Communications*, vol. 15 , no. 5, pp. 56–63, Oct. 2008.

[7] D. Gesbert, M. Shafi, D. Shiu, P.J. Smith, and A. Naguib, "From theory to practice: An overview of MIMO space-time coded wireless systems," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 3, pp. 281–302, Apr. 2003.

[8] B. Hamdaoui and K.G. Shin, "Characterization and analysis of multi-hop wireless MIMO network throughput," in *Proc. ACM MobiHoc,* pp. 120–129, Montreal, Quebec, Canada, Sep. 2007.

[9] Y.T. Hou, Y. Shi, and H.D. Sherali, *Applied optimization methods for wireless networks,* Cambridge University Press, 2014.

[10] IBM ILOG CPLEX Optimizer, software available at http://www-01.ibm.com/software/integration/optimization/cplex-optimizer.

[11] S. Jafar and M. Fakhereddin, "Degrees of freedom for the MIMO interference channel," *IEEE Trans. on Information Theory,* vol. 53, no. 7, pp. 2637–2642, July 2007.

[12] C. Jiang, Y. Shi, Y.T. Hou, S. Kompella, "On the asymptotic capacity of multi-hop MIMO ad hoc networks," *IEEE Trans. on Wireless Commun.,* vol. 10, no. 4, pp. 1032–1037, Apr. 2011.

[13] S.-J. Kim, X. Wang, and M. Madihian, "Cross-layer design of wireless multihop backhaul networks with multiantenna beamforming," *IEEE Trans. on Mobile Computing,* vol. 6, no. 11, pp. 1259–1269, Nov. 2007.

[14] 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 *Proc. IEEE Asilomar Conference on Signals, Systems & Computers (ACSSC),* pp. 122–126, Grove, CA, Oct. 2006.

[15] B. Mumey, J. Tang, and T. Hahn, "Algorithmic Aspects of Communications in Multihop Wireless Networks with MIMO Links," in *Proc. IEEE ICC,* pp. 1–6, Cape Town, South Africa, July 2010.

[16] J. Mundarath, P. Ramanathan, and B.D. Van Veen, "Exploiting spatial multiplexing and reuse in multi-antenna wireless ad hoc networks," *Elsevier Ad Hoc Networks,* vol. 7, no. 2, pp. 281–293, March 2009.

[17] J.-S. Park, A. Nandan, M. Gerla, and H. Lee, "SPACE-MAC: Enabling spatial reuse using MIMO channel-aware MAC," in *Proc. IEEE ICC,* pp. 3642–3646, Seoul, Korea, May 2005.

[18] D. Qian, D. Zheng, J. Zhang, and N. Shroff, "CSMA-based distributed scheduling in multi-hop MIMO networks under SINR model," in *Proc. IEEE INFOCOM,* pp. 2865–2873, San Diego, CA, March 2010.

[19] A. Schrijver, *Theory of Linear and Integer Programming,* Chapter 18, John Wiley & Sons, 1998.

[20] H.D. Sherali and W.P. Adams, *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems,* Chapter 8, Kluwer Academic Publishers, 1999.

[21] Y. Shi, J. Liu, C. Jiang, C. Gao, and Y.T. Hou, "An optimal MIMO link model for multi-hop wireless networks," in *Proc. IEEE INFOCOM,* pp. 1916–1924, Shanghai, China, April 2011 (INFOCOM 2011 Best Paper Runner-Up Award). (Extended version appeared in *IEEE Trans. on Mobile Computing,* vol. 13, no. 7, pp. 1395–1408, July 2014.)

[22] R. Solis, V.S. Borkar, and P.R. Kumar, "A new distributed time synchronization protocol for multihop wireless networks," in *Proc. IEEE Conference on Decision and Control,* pp. 2734–2739, San Diego, CA, Dec. 2006.

[23] Q.H. Spencer, A.L. Swindlehurst, and M. Haardt, "Zero-forcing methods for downlink spatial multiplexing in multiuser MIMO channels," *IEEE Trans. on Signal Processing,* vol. 52, no. 2, pp. 461–471, Feb. 2004

[24] K. Sundaresan, R. Sivakumar, M. Ingram, and T-Y. Chang, "Medium access control in ad hoc networks with MIMO links: Optimization considerations and algorithms," *IEEE Trans. on Mobile Computing,* vol. 3, no. 4, pp. 350–365, Oct. 2004.

[25] K. Sundaresan, W. Wang, and S. Eidenbenz, "Algorithmic aspects of communication in ad-hoc networks with smart antennas," in *Proc. ACM MobiHoc*, pp. 298–309, Florence, Italy, May 2006.

[26] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*, Chapter 7, Cambridge University Press, Cambridge, UK, 2005.

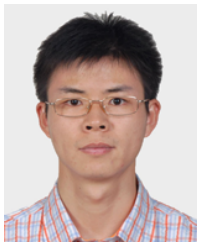
[27] X. Xie, X. Zhang, and K. Sundaresan, "Adaptive feedback compression for MIMO networks," in Proc. of ACM MobiCom, pp. 477–488, Miami, FL, Sep. 2013.

[28] S. Zaks, "Optimal distributed algorithms for sorting and ranking," *IEEE Trans. on Computers,* vol. 5, no. 1, pp. 376–379, April 1985.

[29] H. Zeng, Y. Shi, Y.T. Hou, Rongbo Zhu, and W. Lou, "A Novel MIMO DoF Model for Multi-hop Networks," *IEEE Network,* vol. 28, issue 5, pp. 81–85, Sept./Oct. 2014.

[30] X. Zhang, K. Sundaresan, M.A. Khojastepour, S. Rangarajan, and K.G. Shin, "NEMOx: scalable network MIMO for wireless networks," in Proc. of ACM MobiCom, pp. 453-464, Miami, FL, Sep. 2013.

[31] L. Zheng and D.N.C. Tse, "Diversity and multiplexing: A fundamental tradeoff in multiple-antenna channels," *IEEE Trans. on Information Theory,* vol. 49, no. 5, pp. 1073–1096, May 2003.



Huacheng Zeng (S'09) received his B.E. and M.S. degrees in Electrical Engineering from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2007 and 2010, respectively. He received his Ph.D. degree in Computer Engineering from Virginia Tech, Blacksburg, VA, in 2015. He is currently a Senior System Engineer at Marvell Semiconductor, Santa Clara, CA. He was a recipient of 2014 ACM WUWNET Best Student Paper Award.

Yi Shi (S'02–M'08–SM'13) received his Ph.D. degree in Computer Engineering from Virginia Tech, Blacksburg, VA in 2007. He is currently a Senior Research Scientist at Intelligent Automation Inc., Rockville, MD, and an Adjunct Assistant Professor at Virginia Tech. His research focuses on optimization and algorithm design for wireless networks. He was a recipient of IEEE INFOCOM 2008 Best Paper Award and the only Best Paper Award Runner-Up of IEEE INFOCOM 2011.

Y. Thomas Hou (S'90–M'98–SM'04–F'14) is Bradley Distinguished Professor of Electrical and Computer Engineering at Virginia Tech, Blacksburg, VA. He received his Ph.D. degree in Electrical Engineering from New York University (NYU) Polytechnic School of Engineering in 1998. Prof. Hou's research focuses on developing innovative solutions to complex problems that arise in wireless networks. He is an IEEE Fellow and an ACM Distinguished Scientist. He is the Chair of IEEE INFOCOM Steering Committee and a Distinguished Lecturer of the IEEE Communications Society.

Wenjing Lou (S'01–M'03–SM'08–F'15) is a Professor in the Department of Computer Science at Virginia Tech, USA. She received her Ph.D. degree in Electrical and Computer Engineering from the University of Florida in 2003. Prof. Lou's research interests include cyber security and wireless networks. She is on the editorial boards of a number of IEEE transactions. She is an IEEE Fellow and the Steering Committee Chair of IEEE Conference on Communications and Network Security (CNS). Prof. Lou is currently on IPA assignment as a program director at the US National Science Foundation.

Hanif D. Sherali is a University Distinguished Professor Emeritus in the Industrial and Systems Engineering Department at Virginia Polytechnic Institute and State University. His areas of research interest are in mathematical optimization modeling, analysis, and design of algorithms for specially structured linear, nonlinear, and continuous and discrete nonconvex programs. He is a Fellow of INFORMS and IIE, and an elected member of the US National Academy of Engineering (NAE).

Rongbo Zhu (M'10) is currently a professor in the College of Computer Science of South-Central University for Nationalities, China. He received his Ph. D. degree in communication and information systems from Shanghai Jiao Tong University, China, in 2006. Dr. Zhu was a visiting scholar at Virginia Tech from 2011 to 2012. His research interests are performance optimization and protocol design of wireless networks.

Scott F. Midkiff (S'82–M'85–SM'92) is a professor in the Bradley Department of Electrical and Computer Engineering and currently serves as Vice President for Information Technology and Chief Information Officer at Virginia Tech, Blacksburg, VA, USA. He received his Ph.D. degree in Electrical Engineering from Duke University, Durham, NC. During 2006–2009, he served as a program director at the US National Science Foundation. His research interests include wireless networks, network services for pervasive computing, and cyber-physical systems.