

**Unified Collision-Free  
Coordinated Distributed Scheduling (CF-CDS) in  
IEEE 802.16 Mesh Networks**

Copyright © November, 2008, IEEE. Reprinted from IEEE Transactions On Wireless Communications, Vol. 7, NO. 10, October 2008.

This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of Argon ST's products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to [pubs-permissions@ieee.org](mailto:pubs-permissions@ieee.org).

By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

# Unified Collision-Free Coordinated Distributed Scheduling (CF-CDS) in IEEE 802.16 Mesh Networks

Hua Zhu, *Senior Member, IEEE*, YatKwan Tang, and Imrich Chlamtac, *Fellow, IEEE*

**Abstract**—IEEE 802.16 mesh networking has attracted significant commercial interests recently. In this paper, we propose a unified Collision-Free Coordinated Distributed Scheduling (CF-CDS) algorithm for both control and data transmissions in IEEE 802.16 Mesh mode. Compared to corresponding algorithms defined by the standard [1][3], CF-CDS has two major advantages: (i) CF-CDS truly achieves collision-free control scheduling while the standards CDS algorithm does not, although it claims to; (ii) in addition to periodic scheduling for the control sub-frame, CF-CDS can be applied to on-demand data scheduling in the data sub-frame, where it outperforms the CDS-based 3-way handshakes approach adopted by the standard. We provide performance comparison between standard algorithms and CF-CDS through extensive simulation.

**Index Terms**—IEEE 802.16, WiMax, mesh network, distributed Scheduling, TDMA.

## I. INTRODUCTION

IEEE 802.16 [1][2][3], as one of major competing mesh technologies for metropolitan area networks, has attracted significant interests recently. IEEE 802.16 can operate in either a cellular-like PMP (Point-to-Multipoint) mode or a Mesh mode. The Mesh mode aims to support arbitrary multi-hop communications, which places a challenge on network capacity. As shown by Gupta and Kumar [4], the network capacity decreases substantially as the size (in terms of number of hops) of the network increases. In fact, in many cases, the network capacity is strictly limited by interference. Consequently, how to use the already limited network capacity more efficiently has become one of the most important problems for wireless mesh networks and other types of ad hoc networks. In TDMA-based IEEE 802.16 networks, the design of MAC slot scheduling algorithms has significant impact on performance of individual users (nodes) across the entire network. IEEE 802.16 provides three scheduling algorithms, i.e., centralized scheduling, coordinated distributed scheduling and uncoordinated distributed scheduling. There are tradeoffs among these algorithms, e.g., distributed algorithms are more suitable for multi-hop mesh networks whereas the centralized algorithm may be more attractive for star topology with infrastructure.

Manuscript received April 26, 2007; revised August 18, 2007; accepted September 10, 2007. The editor recommending publication of this paper is D. Wu.

H. Zhu and Y.-K. Tang are with the San Diego Research Center, San Diego, CA, 92121 (e-mail: {hua.zhu, david.tang}@sdrcinc.net).

I. Chlamtac is with Create-Net Research Consortium, Italy (e-mail: chlamtac@create-net.org).

Digital Object Identifier 10.1109/TWC.2008.070435.

Given our focus on mesh networking, the scope of this paper is limited to the coordinated distributed scheduling (CDS) algorithm.

We found that the first weakness of CDS is its claim of “collision-free” transmissions. CDS is based on an easily overlooked modeling approximation that interference in wireless networks can be modeled via wired-like logical connectivity graphs, which we term quasi-interference modeling. Moreover, it adopts an open-loop scheduling approach, i.e., each node speculates the schedules of other nodes merely based on a commonly-agreed random generator. As a result, although CDS can successfully achieve “collision-free” scheduling in quasi-interference environments, our simulations clearly show that it suffers substantial collisions in realistic environments. This will significantly degrade the performance of 802.16 mesh networks. More details are elaborated in Section 3.

The second weakness of CDS is that it only provides a platform for periodic control message transmissions. The scheduling for data transmissions are to be resolved by traditional three-way handshakes occurred between node pairs piggybacked in their control messages. Although the standard provides signaling message formats, the detail algorithms are left unstandardized (please refer to Section 5.1 for more details).

To address both weaknesses of CDS, we propose a unified Collision-Free Coordinated Distributed Scheduling (CF-CDS) algorithm.

- For control scheduling: as the name indicates, the baseline CF-CDS achieves efficient collision-free TDMA scheduling based on a very different close-loop paradigm of node coordination. Our extensive simulation results clearly show that, in realistic environments, CF-CDS successfully achieves true collision-free transmissions while CDS fails to do that. Comparison on scheduling intervals and communications overhead are also provided.
- For data scheduling: the enhanced CF-CDS performs on-demand scheduling for generic data traffic to improve scheduling latency and TDM resource allocation efficiency. Simulation results show that CF-CDS outperforms the three-way handshakes approach in both delay and throughput.

It is worth noting that, although CF-CDS chooses a very different scheduling approach, the required modifications to existing 802.16 message format are minimal. The only necessary change is to replace all scheduling fields in CDS with a

single scheduling bitmap required by CF-CDS.

The remainder of the paper is organized as follows. An overview of IEEE 802.16, Mesh mode and CDS in specific are described in Section 2. In Section 3, we address the quasi-interference modeling that makes CDS fail to achieve collision-free scheduling in realistic scenarios, which is demonstrated by simulations. In Section 4 and 5, we propose baseline and enhanced CF-CDS for control and data scheduling, respectively. After summarizing related work in Section 6, we conclude the paper in Section 7.

## II. OVERVIEW OF IEEE 802.16, MESH MODE, AND CDS

### A. Background of IEEE 802.16

The IEEE 802.16 [1][2][3] standard is originally emerging as a promising fixed broadband wireless technology to finally resolve the “last mile” Internet access in conjunction with IEEE 802.11. The IEEE 802.16 is to provide broadband access of up to 75 Mbps and typical base station infrastructure coverage of 5 miles in PMP (Point-to-Multipoint) operation mode, which is much greater than the coverage of 802.11. In addition, the IEEE 802.16e has also been approved on Dec. 7th 2005 to support Mobile WirelessMAN.

IEEE 802.16-2004 supports four PHY specifications for metropolitan area networks in the licensed bands (MMDS and ETSI), which are WirelessMAN-SC (SingleCarrier), -SCa, OFDM, OFDMA, as well as their counter parts (-SCa, -OFDM, and -OFDMA) for the unlicensed bands WirelessHUMAN (Wireless High-Speed Unlicensed Metropolitan Area Networks). Most of PHYs are designed for NLOS operations in the frequency bands below 11 GHz, except SC, which is for operations in the 10-66GHz bands, and it supports both TDD and FDD operations.

IEEE 802.16 MAC supports two modes: PMP and Mesh, respectively. The former organizes nodes into a cellular like structure consisting of a base station (BS) and subscriber stations (SS). The channels are divided into uplink (from SS to BS) and downlink (from BS to SS), both shared among the SSs. PMP mode requires all SSs to be within the transmission range of the BS. In PMP mode, traffic only occurs between the BS and SSs. On the other hand, in the mesh mode, an ad hoc network is formed with all nodes acted as relaying routers in addition to their sender and receiver roles, although there may still be nodes to serve as BSs and provide backhaul connectivity. In order to achieve efficient multi-hop data transmissions, the Mesh mode defines three scheduling schemes, i.e., centralized, coordinated distributed, uncoordinated distributed scheduling, which are used by control messages for various functionalities, such as to grant net entry requests, topology discovery and management, and to schedule data transmissions. Except uncoordinated distributed scheduling, which is designed to support low duty-cycle traffic scenarios with simple unreliable control message transmissions, the other two scheduling algorithms are designed to provide “collision-free” transmissions of control messages.

### B. General Description of IEEE 802.16 Mesh Mode

802.16 offers both PMP and Mesh modes. The main difference between PMP and Mesh modes is that in the PMP

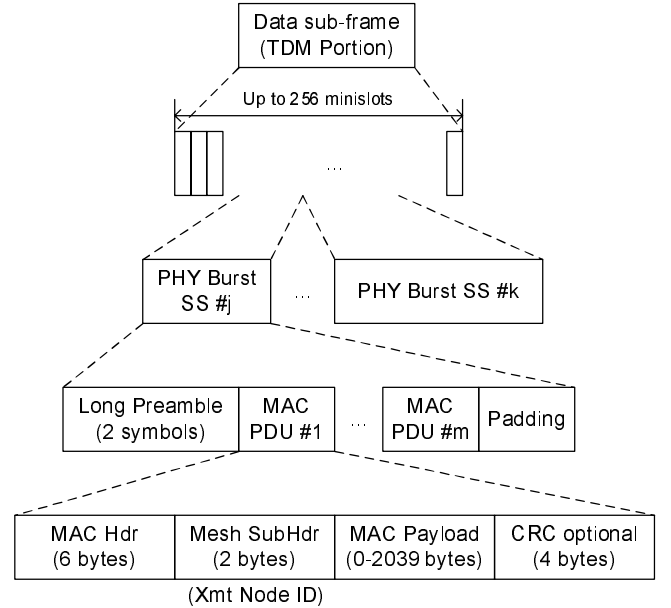


Fig. 1. Mesh frame structure.

mode, traffic only occurs between the BS (Base Station) and SSs (Subscriber Stations), while in the Mesh mode, traffic can be relayed via other SSs, and also can occur directly between SSs. It is worth noting that a concept of relay stations (RSs) and tree-based mobile multi-hop relay (MMR) networking has also been introduced for PMP mode, which may expand the coverage area and enhance throughput [5][6] of the PMP mode. Comparing with the moderate tree-based multi-hop network topology of 802.16mmr, 802.16 mesh mode places more challenges on the link scheduling algorithms. In order to achieve efficient collision-free multi-hop data transmissions, the Mesh mode defines three scheduling schemes, i.e., centralized, coordinated distributed, uncoordinated distributed scheduling, to resolve wireless interference occurred in 2-hop (or 3-hop) neighborhood of a node. According to the specification, 3-hop extended neighborhood can be used in the environment that is close to free-space. Figure 1 shows the frame structure of the Mesh mode. Each frame is divided into two parts: i) a control sub-frame consisting of  $MSH\_CTRL\_LEN$  (0-15) transmission opportunities (termed  $XmtOps$ ) and ii) a TDM data sub-frame consisting of up to 256 minislots. For high reliability, all the  $XmtOps$  are in fixed length of 7 OFDM symbols ( $T_S$ ). The duration of a minislot is often of 4 OFDM symbols. The number of minislots can be derived given the frame duration ( $T_F$ , 2.5-20ms),  $XmtOp$  duration, minislot duration, and the number of control  $XmtOps$ . There are two types of control sub-frame, i.e., schedule and network. The latter reoccurs once every  $Scheduling\_Frames$ , providing basic functionality of network entry and topology management. The former is to resolve the transmission schedule of data sub-frame. A node will not transmit in any minislot that is not reserved for its use. While reserved, each PHY burst may take multiple minislots, which is shown in Figure 2.

The network control sub-frame serves primarily for new nodes to gain access to the network. In each network control

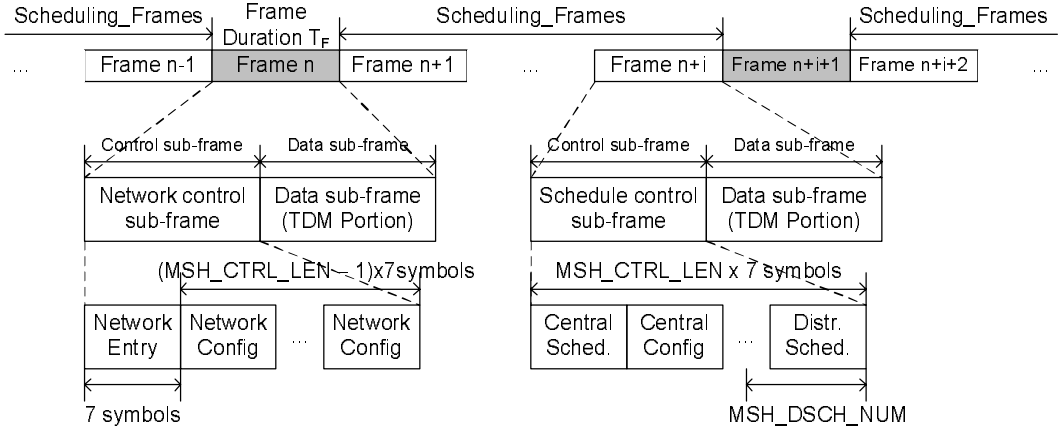


Fig. 2. Data sub-frame structure.

sub-frame, the first  $XmtOp$  is dedicated to network entry message (MSH-NENT) via unreliable contention access. A successful network entry relies on additional handshakes between the new node and the sponsor node. The remaining  $(MSH\_CTRL\_LEN - 1) XmtOps$  are dedicated to network configuration messages (MSH-NCFG). The schedule control sub-frame is used to schedule minislots for TDM transmissions in the data sub-frame. However, the access of  $XmtOps$  in the schedule control sub-frame itself requires certain rules, which could be centralized, distributed, or a combination of both scheduling methods. If both scheduling methods co-exist, the first  $(MSH\_CTRL\_LEN - MSH\_DSCH\_NUM) XmtOps$  within the control sub-frame are allocated for centralized scheduling messages (MSH-CSCH/MSH-CSCF), while the remaining ones are for distributed scheduling messages (MSH-DSCH).

### C. Coordinated Distributed Scheduling (CDS)

Let us first give a general picture of all specified scheduling algorithms. The centralized scheduling relies on a centralized coordinator, i.e., Mesh BS (MBS), to schedule MSH-CSCH/MSH-CSCF packets in a collision-free manner, which is the best for links supporting persistent data traffic streams. The distributed scheduling can be further divided into coordinated distributed scheduling and uncoordinated distributed scheduling. The former, which is used by MSH-NCFG and MSH-DSCH, accesses  $XmtOps$  in a “collision-free” manner. On the other hand, the latter, which can be used only by MSH-DSCH, adopts a simple contention approach in which collisions may occur if multiple nodes are transmitting in the same  $XmtOp$ . Uncoordinated distributed scheduling is only best for links with occasional or brief traffic needs. Coordinated distributed scheduling, which is designed for general Mesh operations with substantial peer-to-peer traffic needs, is the main focus of this paper. Due to the limit of space, we will not elaborate other scheduling algorithms in this paper. Interested readers may refer to standard specification [1] and the tutorial for more details on Mesh model [7].

Coordinated distributed scheduling (CDS) is designed to achieve collision-free periodical transmissions for two types of control messages, i.e., MSH-NCFG and MSH-DSCH, respectively. Since the exact same algorithm is used independently

for these two types of messages in separated  $XmtOps$ , we can simply analyze the behaviors of one, and the result is applicable to the other.

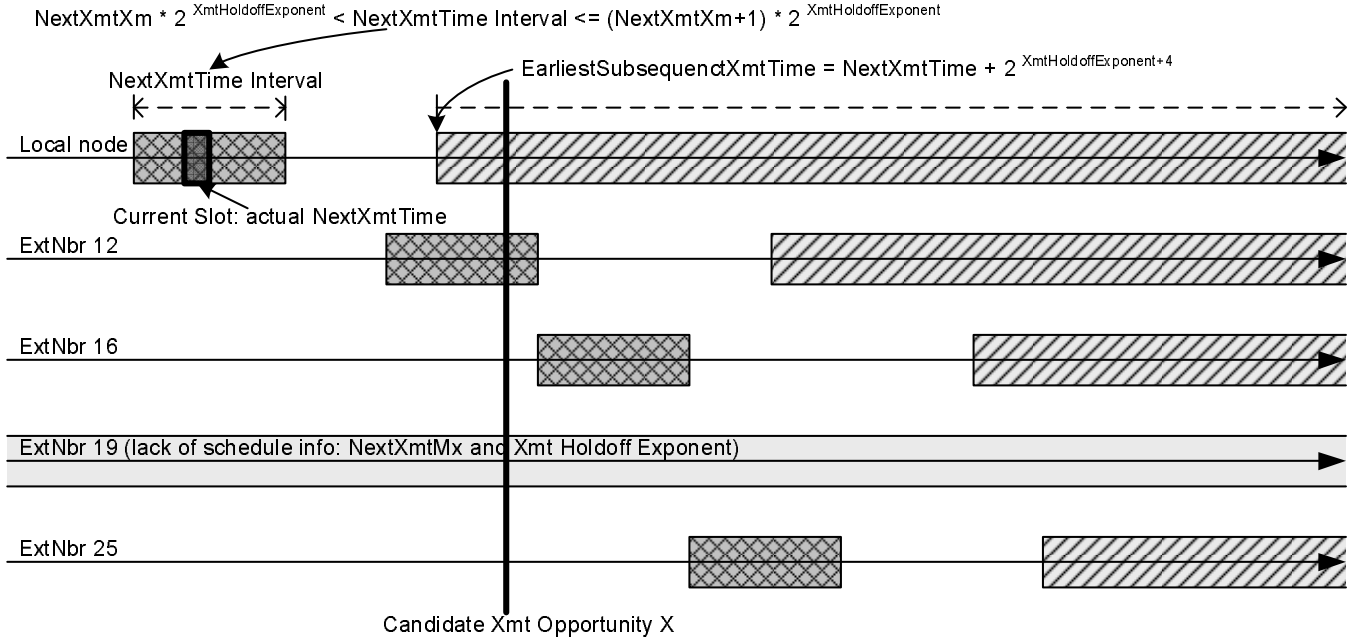
The general concept of CDS is to let nodes running the scheduling algorithm independently derive pseudo-random but predictable behaviors by exchanging 2- or 3-hop neighborhood schedule information with each other. Both the randomness and predictability are achieved by dynamically constructing random generator seeds for each node according to a common rule. The seed for a node is constructed based on its unique node ID and the index (or timestamp) of a candidate  $XmtOp$ . Given the neighborhood information, the random number generated locally will be the same with the corresponding one generated at a neighboring node. Therefore, predictability is achieved. In detail, by using previously scheduled transmission opportunities, nodes compute and exchange their next collision-free  $XmtOps$ , as well as any available schedules of their 2- or 3-hop neighbors, which is in the format of two scheduling related parameters (i)  $NextXmtXm :: 5bits$  and (ii)  $XmtHoldoffExponent :: 3bits$ .

Given these two parameters of a specific neighbor, a node can determine a bounded interval for  $NextXmtTime$  as well as  $EarliestSubsequentXmtTime$  of the neighbor as the following:

$$NextXmtXm2^{XmtHoldoffExponent} < NextXmtTime \\ \leq (NextXmtXm + 1)2^{XmtHoldoffExponent} \quad (1)$$

$$EarliestSubsequentXmtTime = \\ NextXmtTime + 2^{XmtHoldoffExponent+4} \quad (2)$$

Since the exact scheduled  $XmtOp$  of the neighboring node is unknown, as an implementation issue, one may define  $NextXmtTime$  to be the last  $XmtOp$  within the interval when calculating  $EarliestSubsequentXmtTime$ . As shown in Figure 3, at the previously scheduled  $XmtOp$ , a node (namely the local node) will run an election algorithm to find its next collision-free  $XmtOp$ . Based on the calculated  $NextXmtTime$  time interval and  $EarliestSubsequentXmtTime$ , a node can exclude a subset of neighboring nodes (e.g., Nbr 16 and 25 in Figure 3) from the competing neighbors of a particular candidate  $XmtOp$  X, which reduces the number of unnecessary idleness in the schedule and improves the utilization of  $XmtOps$ . For



Nodes 12 and 19 are eligible competing neighbors for Candidate Xmt Opportunity X

Fig. 3. Coordinated distributed scheduling (CDS) Election-based approach.

a particular candidate  $XmtOp$  starting from the  $EarliestSubsequentXmtTime$  of a local node, if it generates the largest random number amongst all the eligible competing nodes, it wins this  $XmtOp$ . Otherwise, it will keep incrementing the candidate  $XmtOp$  and running the same election algorithm until it wins the election [7].

### III. COLLISIONS DUE TO QUASI-INTERFERENCE MODELING APPROXIMATION

Although CDS intends to provide collision-free transmissions by resolving collisions incurred by concurrent transmissions in a 2-hop or 3-hop neighborhood, it does rely on an easily overlooked modeling approximation, a simplified quasi-interference RF model in which wireless networks are approximated with wired-like logical connectivity graphs. In such a model, collisions will not take place at the receiver if there is no connectivity edge to link the receiver to the concurrent senders. It appears as if RF signals are completely cut-off beyond all 1-hop neighbors with zero energy propagated further. Evidently, this idealized assumption cannot be held true in a realistic environment, where interference accumulated from multiple senders far away may damage a transmission from a direct neighbor of the receiver.

Aware of this modeling discrepancy, CDS adopts a “quick fix” in that scheduling can be done in extended 3-hop neighborhood (specified by  $ExtendedNeighborType$ ) when necessary, with more overhead. However, since the paradigm of distributed node coordination has not been changed, collisions cannot be fundamentally eliminated. Furthermore, unconditionally applying 3-hop scheduling to all the senders makes CDS less efficient in terms of spatial re-use since some of them may actually operate well with 2-hop scheduling. Nevertheless, based on our extensive simulations, we find that

the collisions involved in CDS are substantial, even for 3-hop scheduling in certain non-free-space-like RF environments, as shown in the result below. Evidently, the high percentage of collisions will significantly impair the performance of 802.16 MAC since many control messages, which rely on CDS to access the channel, may not be received by certain neighbors, if not all of them. Moreover, since the scheduling of data transmissions relies on the control message handshakes, the actual performance experienced by upper layer will be varied significantly, if not unpredictable. Extensive simulations have been conducted to evaluate the performance of CDS. We implement the IEEE 802.16 Mesh Mode in QualNet [8], with major configurations listed in Table I. We adopt 802.16 PHY and MAC profiles defined in [1]. Simulation scenarios include both “dense deployment with high data rate” and “sparse deployment with low data rate”. For the former type of scenarios, the radio range is limited given the 64-QAM 3/4 modulation. For the latter, longer radio range is achieved with QPSK 1/2 modulation. Various combinations of propagation models and multi-hop topologies with 49 nodes are constructed.

Identical  $XmtHoldoffExponent$  is used in this paper. However, consistent results have been observed for non-identical  $XmtHoldoffExponent$  cases. The following set of performance metrics is collected:

- Reception collision ratio (%): overall collision ratio seen at receivers (only collisions caused by interference from concurrent transmissions are counted)
- Scheduling interval ( $XmtOp$ ): overall average number of corresponding transmission opportunities between two consecutive scheduled slots
- Number of extended neighbors: overall average number of neighbors in  $n$ -hop extended neighborhood

TABLE I  
SIMULATION CONFIGURATION

<b>PHY Profile (dense, high data rate): ProfP3_7: WirelessMAN-OFDM PHY profile for 7 MHz channelization</b>	
Modulation	64-QAM 3/4
Carrier frequency	3.5GHz ETSI licensed band
Channel bandwidth	7 MHz
$E_b/N_0$ (for $BER < 10^{-6}$ )	19.0 dB
OFDM raw bitrates	21.60 Mb/s (for $T_g = T_p/4$ )
Receiver SNR threshold	23.894 dB
Noise power	-97.073 dBm (2.8027e-17 mW/Hz)
Receiver power sensitivity threshold	-73.179 dBm (minimum performance requirement -69 dBm)
Transmission Power	34.77 dBm (3W)
Antenna Height	1.5m
<b>PHY Profile (sparse, low data rate): ProfP3_7: WirelessMAN-OFDM PHY profile for 7 MHz channelization</b>	
Modulation	QPSK 1/2
Carrier frequency	3.5GHz ETSI licensed band
Channel bandwidth	7 MHz
$E_b/N_0$ (for $BER < 10^{-6}$ )	10.5 dB
OFDM raw bitrates	4.80 Mb/s (for $T_g = T_p/4$ )
Receiver SNR threshold	9.4 dB
Noise power	-97.073 dBm (2.8027e-17 mW/Hz)
Receiver power sensitivity threshold	-87.673 dBm (minimum performance requirement -84 dBm)
Transmission Power	34.77 dBm (3W)
Antenna Height	3.0m
<b>MAC Profile: ProfM3_Mesh: WirelessMAN-OFDM Basic Packet Mesh MAC profile</b>	
Frame duration	code {0x01} $T_F = 4\text{ms}$
Symbol duration	$T_S = 3.7037 \mu\text{s}$
Minislot size	1PS = 4 symbols
Control opportunity size	7 symbols
Scheduling_Frames	8
MSH_CTRL_LEN	8
MSH_DSCH_NUM	8 (all for coordinated distributed scheduling)
Number of minislots	256
<b>Simulation Scenarios</b>	
QAM $\frac{3}{4}$ + Free space model + 49 nodes in 7x7grid with a grid distance of 300 m	
QAM $\frac{3}{4}$ + Free space model + 49 nodes uniformly distributed in 2.5kmx2.5km	
QAM $\frac{3}{4}$ + Two ray model + 49 nodes uniformly distributed in 2.5kmx2.5km	
QAM $\frac{3}{4}$ + ITM model + 49 nodes uniformly distributed in 1.1kmx1.1km mountainous terrain with relative elevation range of 200 m. ITM is a terrain based propagation model	
QPSK $\frac{1}{2}$ + Free space model + 49 nodes in 7x7grid with a grid distance of 2.7km	
QPSK $\frac{1}{2}$ + Free space model + 49 nodes uniformly distributed in 20kmx20km	
QPSK $\frac{1}{2}$ + Two ray model + 49 nodes uniformly distributed in 20kmx20km	
QPSK $\frac{1}{2}$ + ITM model + 49 nodes uniformly distributed in 10kmx10km mountainous terrain	

In the simulation, we can identify error receptions caused by interference from concurrent transmissions (i.e., received  $SINR < SNR$  threshold, but received  $SNR \geq SNR$  threshold, and received power  $\geq$  receiver power sensitivity). Note that the receiver power sensitivity and SNR threshold have been carefully configured based on bit-energy-to-noise-density ratio ( $E_b/N_0$ ) so that reception error will not occur if the only source of interference is local thermal noise. Therefore, reception error implies external interference, or reception collision. Also note that we only collect statistics in stabilized condition, and therefore we can exclude collisions in transient condition. Figure 4 shows the non-zero reception collision

ratio, which verifies our analysis in this section. With the smallest  $XmtHoldoffExponent$ , nodes have high possibility to schedule concurrent transmissions beyond 2- or 3-hop scheduling neighborhood. Based on the idealized quasi-interference model, such spatial re-use and concurrent transmissions should not cause any collision. However, we observe substantial amount of collisions for both dense and sparse deployment. Specifically, the collision ratios are up to 20.78% (2-hop scheduling) and 6.76% (3-hop) in dense deployment, and 9% (2-hop) and 5% (3-hop) for sparse deployment, respectively. The reception collision ratio decreases when  $XmtHoldoffExponent$  increases from 0 to 4, for extend neighborhood type

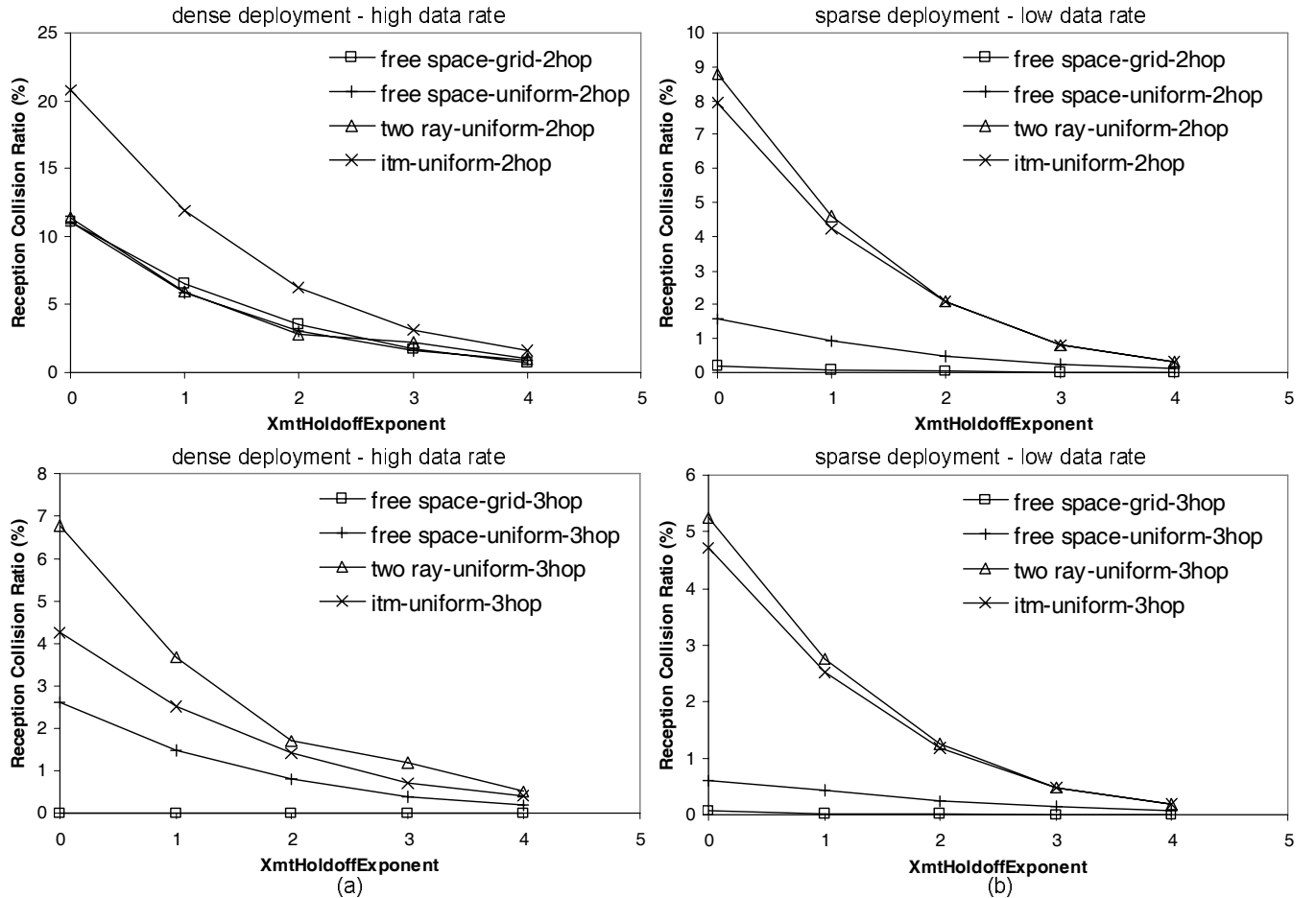


Fig. 4. CDS reception collision ratio vs.  $XmtHoldoffExponent$  (a) Dense deployment with high data rate (64QAM 3/4), (b) sparse deployment with low data rate (QPSK 1/2)

of 2 and 3, respectively. Note that it decreases quite fast when  $XmtHoldoffExponent$  increases from 0 to 1 and from 1 to 2, but slower for larger  $XmtHoldoffExponents$  (3 and 4). This is because, for the latter, the holdoff interval (128 and 256, respectively) is substantially larger than the number of nodes in the affected interference neighborhood, even larger than the total number of nodes in the network. As a result, the probability of concurrent transmissions has already been significantly “diluted”, and keeping increasing the holdoff interval does not benefit much more. This insight is useful for selecting appropriate  $XmtHoldoffExponent$  to alleviate collisions. However, increasing  $XmtHoldoffExponent$  also results in longer scheduling interval, leading to large turn around time for data scheduling handshakes. Furthermore, since each successful data scheduling handshake has to be achieved by successful transmissions of multiple control messages, the compound failure rate of handshakes is even more significant than the collision ratio of individual control messages. Figure 5 shows the overall average schedule interval with respect to  $XmtHoldoffExponent$ , for 2/3-hop neighborhood type, respectively. Since for most cases in our simulation (except for  $XmtHoldoffExponent = 0$  in 3-hop extended neighborhood), the number of neighbors in the extended neighborhood is equal to or smaller than the holdoff interval, therefore, the resulted schedule interval is close to the lower bound specified by the

holdoff interval (i.e., the dot lines in Figure 5), which is equal to  $2^{XmtHoldoffExponent}$ .

#### IV. BASELINE CF-CDS FOR CONTROL SUB-FRAME SCHEDULING

Based on the analysis and simulations in Section 3, we recognize that the present CDS algorithm in IEEE 802.16 fundamentally cannot guarantee collision-free scheduling. Note that the collisions we observed are not due to transient states while the algorithm is converging. Instead, those collisions will always exist with certain statistical probability depending on the network topology and RF environment. The main reason is that, CDS solely relies on the predictable pseudo random generator at each node to independently scheduling  $XmtOps$ . Such type of node coordination paradigm can only prevent quasi-interference collisions, but it lacks necessary means to detect collisions due to non-quasi-interference.

To solve this problem we propose a better alternative, i.e., CF-CDS, which follows a very different paradigm in terms of how nodes actively interact and coordinate with each other. Although CF-CDS leverages the concepts of quasi-interference approximation and logical connectivity graph, it adopts a more proactive collision monitoring and adaptation mechanism. Therefore, by using CF-CDS, all collisions, regardless whether caused by direct neighbors of the receiver

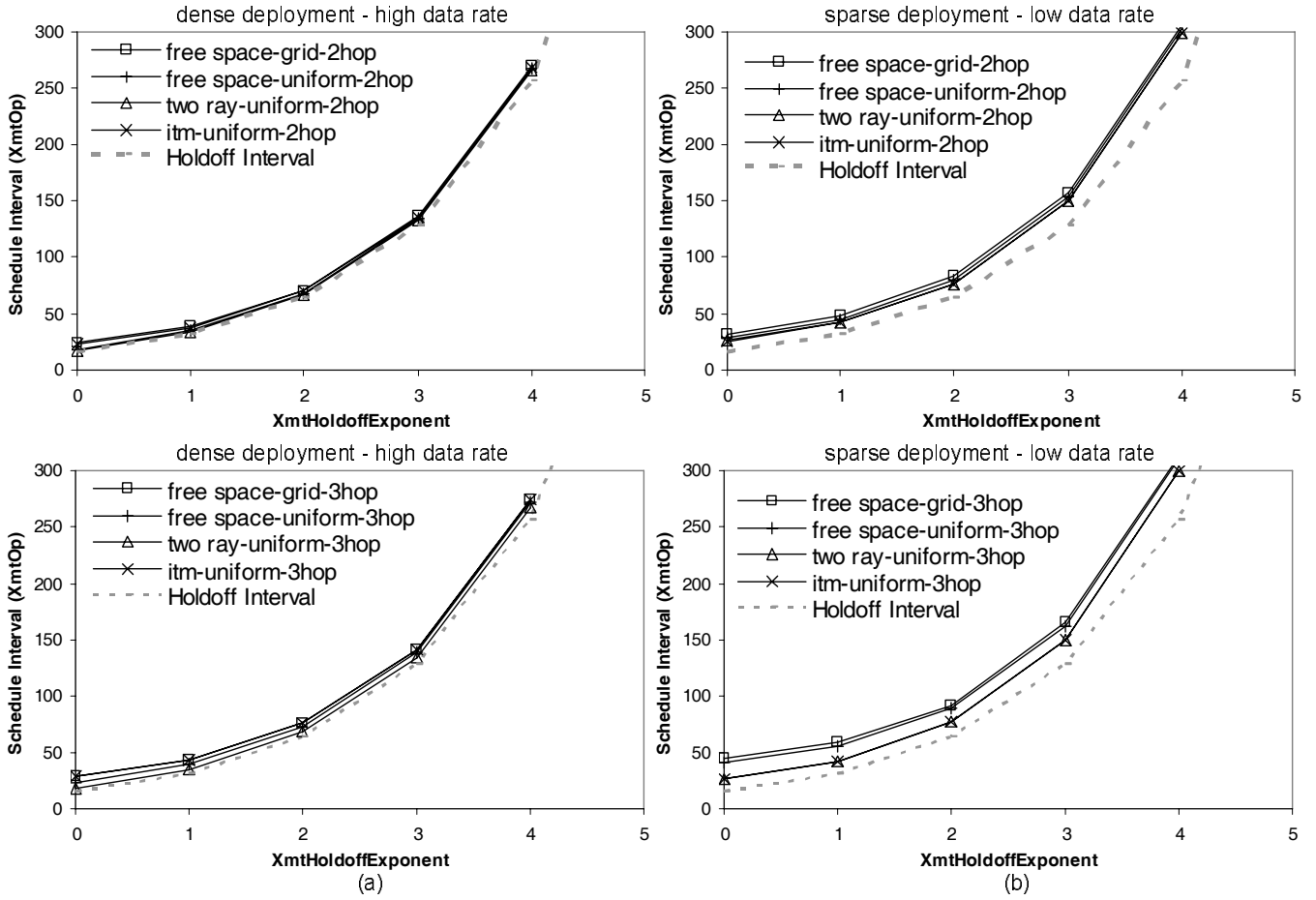


Fig. 5. CDS schedule interval vs.  $XmtHoldoffExponent$  (a) Dense deployment with high data rate (64QAM 3/4), (b) sparse deployment with low data rate (QPSK 1/2)

(i.e., quasi-interference) or accumulative interference other than neighbors (non-quasi-interference), are all tracked and used by the sender to adjust the schedule accordingly. In this way, CF-CDS maintains a high scheduling efficiency by maximizing spatial re-use beyond 2-hop as much as possible, but it is also capable of adjusting schedule appropriately if collisions due to non-quasi-interference beyond 2-hop are detected. Unlike CDS, collisions only occur in the transient state of CF-CDS.

Additionally, given the recent development of IEEE 802.16e, the perspective of mobile metropolitan area mesh networks may soon attract a lot of interests. While CDS is not designed to support mobility scenarios, CF-CDS is robust and can be directly applied to mobile mesh networks without any modification. This is because that, in CF-CDS, nodes promptly react to the outcome of all changes, including slot schedule changes as well as neighboring topology changes due to mobility, in the exactly same way of close-loop monitoring. On the other hand, CDS has to resolve the additional problems such as how to carefully balance the neighbor lifetime and node mobility. Since CDS is based on open-loop operations, the mismatch between mobility and algorithm configuration will either further introduce excessive collisions in addition to the residual collisions shown in static scenarios, or incur inefficiency in slot utilization. Nevertheless, the mobility factor is out of the scope of this paper.

#### A. Baseline CF-CDS Algorithm

In CDS, each signaling message includes 8 bits scheduling information for itself and 8 bits per its neighbor. In CF-CDS, each signaling message includes a scheduling bitmap (named “Transmitted Bitmap”, or TB) of a pre-configured CF-CDS scheduling cycle. Each node running CF-CDS periodically sends one message per scheduling cycle. The number of  $XmtOps$  ( $N_{cycle}$ ) per scheduling cycle closely approximates the role of  $XmtHoldoffExponent$  in CDS. Only 1 bit information per  $XmtOp$  is needed, therefore scheduling cycle is of  $N_{cycle}$  bits. In addition to the TB, each node keeps an addition  $N_{cycle}$ -bit long Local Bitmap (LB) to track accumulative scheduling information provided by the scheduling bitmaps inside received messages from its neighbors. To distinguish all the received TBs from the TB a node itself sends out, we denote the former RBs in the context of receivers thereafter. With two bitmaps (TB and LB), nodes can differentiate direct neighbors from indirect neighbors, which prevent flooding scheduling information of indirect neighbors across the entire network and end up with no spatial re-use. Two more local variables, namely  $myXmtOp$  and  $ifTx$  respectively, are required by CF-CDS.  $myXmtOp$  stores the scheduled  $XmtOp$  of the node (with “-1” stands for “unscheduled”).  $ifTx$  is used for conflict (collision) detection. The pseudo code and graphic illustration of CF-CDS work flow are shown in Figure 6 and

```

Line1 (curXmtOp and myXmtOp are wrt. XmtOp index)
Line2
Line3 Init{
Line4   TB = 0; // array size =  $N_{cycle}$ 
Line5   LB = 0; // array size =  $N_{cycle}$ 
Line6   myXmtOp = -1; // my reserved slot
Line7   ifTx = false;
Line8   counter = 0; // number of conflicting neighbors
Line9   curXmtOp = -1;
Line10  set XmtOp timer to zero delay
Line11 }
Line12
Line13 ProcessXmtOpTimerAtSlotBeginning{
Line14   curXmtOp ++;
Line15   curXmtOp %  $N_{cycle}$  ;
Line16   set XmtOp timer to a XmtOp duration;
Line17   If (myXmtOp = -1) AND (have listened for one cycle after bootup or net entry){
Line18     random pick myXmtOp among (myXmtOp AND all other i's that LB[i]=0);
Line19   }
Line20   else if (curXmtOp = myXmtOp) AND (counter>0) {
Line21     random pick myXmtOp among (myXmtOp AND all other i's that LB[i]=0);
Line22     counter = 0;
Line23     ifTx = false
Line24   }
Line25   if (myXmtOp) {
Line26     TB[myXmtOp] = 1;
Line27     Transmit TB;
Line28     TB = 0; // the whole array is reset
Line29     LB = 0; // the whole array is reset
Line30     ifTx = true;
Line31   }
Line32 }
Line33 Recv_Message_with_RB{
Line34   TB[curXmtOp] = 1;
Line35   // OR for al elements in the array, RB is the received TB
Line36   LB = LB OR RB;
Line37   // Collision Check
Line38   If (RB[myXmtOp] = 0) AND (ifTx = true)
Line39     counter++;
Line40 }

```

Fig. 6. CF-CDS algorithm pseudo code

7, respectively.

On the sender side, a node will send out its own *TB* in its previously scheduled *XmtOp*. In case that *myXmtOp* = -1, which occurs during initialization or convergence phase, a node will randomly pick an idle *XmtOp* based on its latest *LB* information. On the receiver side, when a node receives a *RB*, it updates its *TB* and *LB* with different update rules. For *TB*, it only sets the bit associated with the current *XmtOp* in which it received the *RB* to 1 (refer to Line 34 in Figure 6), which reflects the nodes direct neighbor environment. For

*LB*, it performs an OR operation with the *RB* (Line 36 in Figure 6), which reflects the accumulative environment. "1" in the *LB* means the corresponding *XmtOp* is taken by an interferer and thus ineligible for the owner of the *LB* during scheduling. CF-CDS is memoryless in that it only keeps one-cycle scheduling information. Therefore, no additional signaling overhead is needed if any rescheduling occurs. Note that the memoryless (i.e. persistence level = 1 frame) scheduling is a design decision. Optionally, larger persistence level can be used in CF-CDS. However, based on our simulation results,

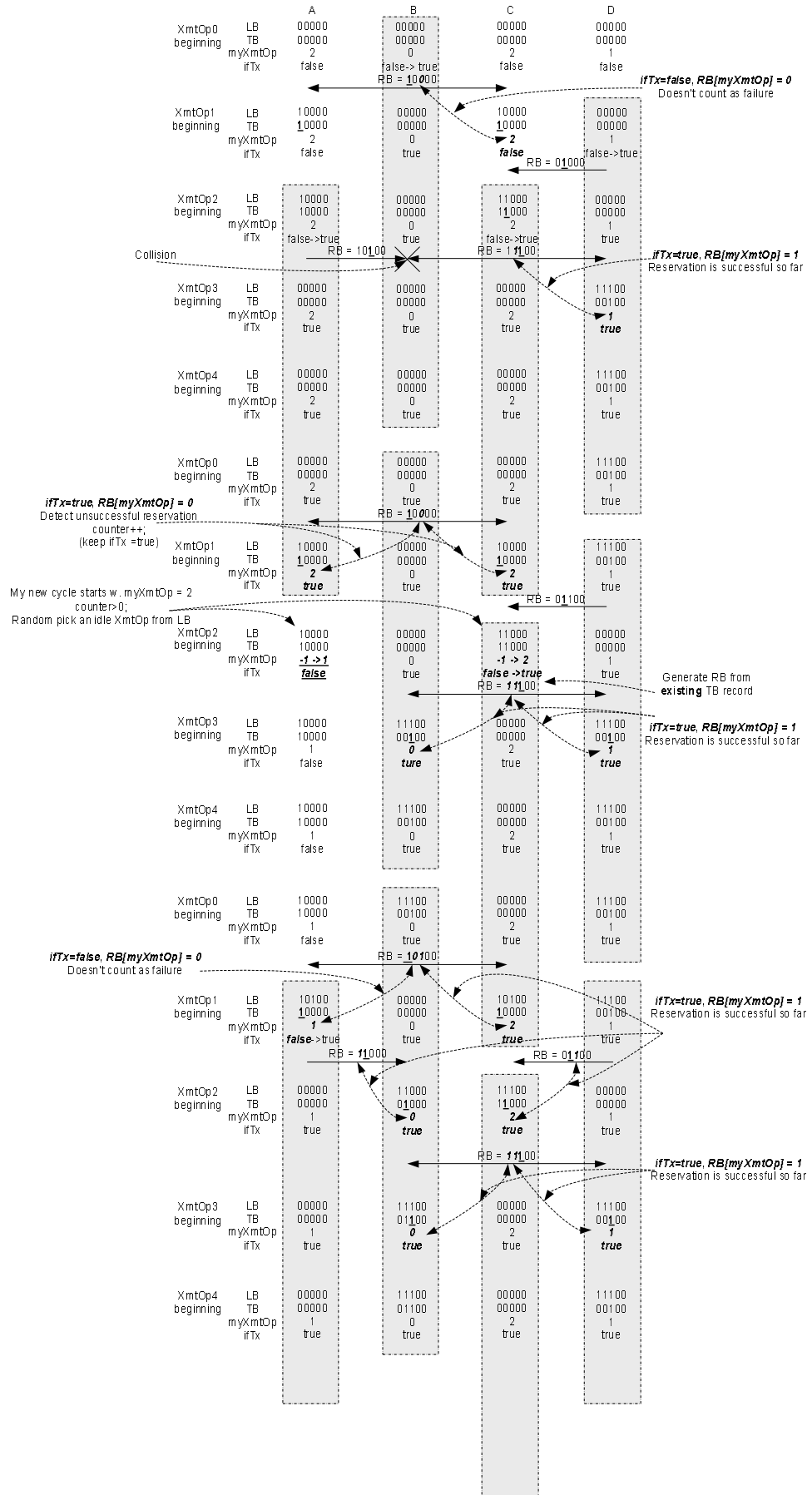


Fig. 7. Graphic illustration of CF-CDS algorithm

memoryless operation does not incur instability, which makes it a good choice considering other design tradeoffs, including responsiveness and overhead.

Collision detection is also performed upon the reception of a RB. Inside a RB, if the bit corresponding to  $myXmtOp$  is not shown as “1” and  $ifTx = true$ , a node detects reception failure of its previous transmission at the originator of RB (Line 38 in Figure 6). Note that the reception failure can be caused by 1-hop collision at the receiver, or accumulative interference multi-hop away from the receiver. The latter factor is the fundamentally undetectable case in CDS. A node counts the number of such conflicts for an entire cycle, and then decides if re-scheduling is needed. Current implementation of CF-CDS (Line 20 in Figure 6) enforces a zero-tolerance of such reception failure, which means it strictly ensures the successful delivery of its scheduled message to all its 1-hop neighbors. Nevertheless, CF-CDS is generic and can be customized with non-zero thresholds if desired. Furthermore, an even more powerful decision component can be developed to differentiate receivers such that the scheduling algorithm transforms from broadcast algorithm to unicast/multicast oriented algorithm. Finally, with a small possibility, a node may miss a collision if another simultaneous transmission succeeds due to capture effects. A solution for this issue, found in WiMedia [15], is to further include an owner indicator associated with the RB, which of course increases the overhead. As a design tradeoff, instead of using 16-bit connection ID, nodes may use a shorter randomly hashed field to alleviate capture effect. Nevertheless, according to our extensive simulations, the impact of capture effect is not significant.

### B. Simulation Results

Figure 8 shows the performance comparison between CDS and CF-CDS in one scenario (free space model + 49 uniformly distributed nodes), with consistent results for other scenarios. In the top figure, it is clear that CF-CDS incurs zero collision for various holdoff intervals except for the cases of 16 and 32  $XmtOps$ . For these two cases, CF-CDS only averages negligible 6.51 and 1.41 collisions per node for holdoff interval of 16 and 32, respectively during the entire simulation, which all happened during the initial convergence phase. By using CF-CDS, nodes can intelligently detect collisions caused by accumulative interference beyond any artificial hop boundary imposed by quasi-interference model. And it only takes a few rounds of trials before all schedules converge. When the number of neighbors is close to the holdoff interval, the algorithm takes more rounds to converge. Of course, if the number of neighbors is greater than the holdoff interval, there will be nodes being blocked without any  $XmtOp$  to schedule a transmission. Proper configuration of holdoff interval usually should be larger than the maximum number of 2-hop neighbors per nodes.

In the middle figure, nodes with CF-CDS schedule transmissions at a constant pace of the exact configured holdoff interval (except during the convergence), while the average schedule interval of CDS consists of extra amount of  $XmtOps$ , which may depends on the number of neighbors of a node.

In the bottom figure, the communications overhead in terms of the length of specific scheduling fields in control

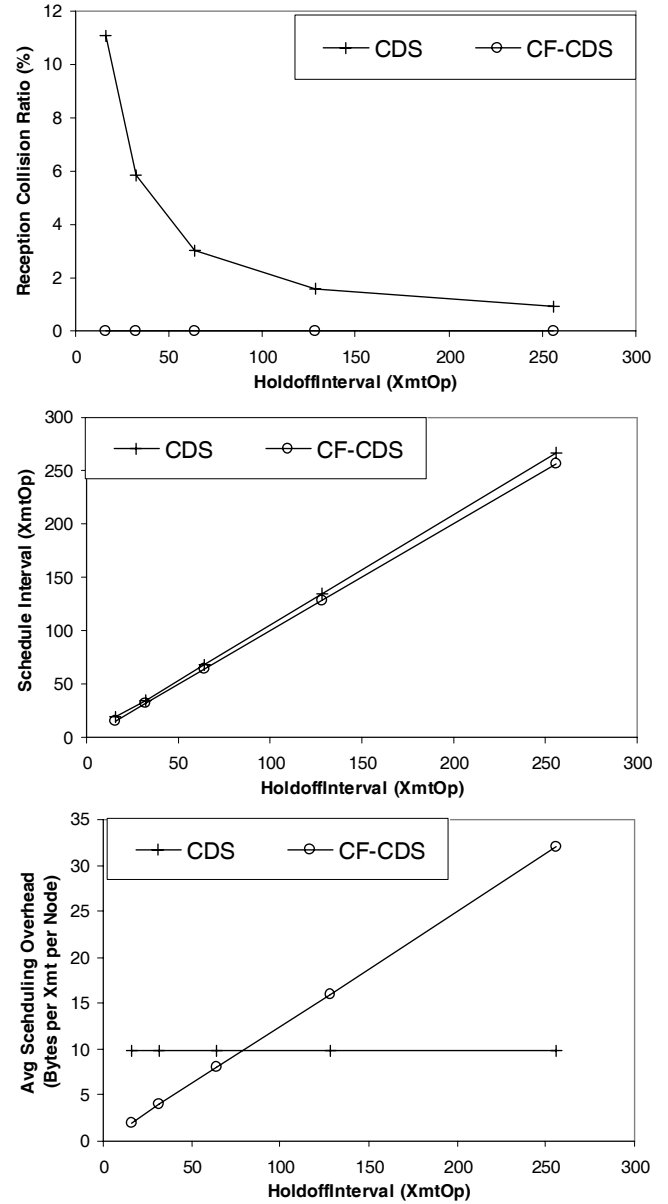


Fig. 8. Performance comparison between CDS & CF-CDS

messages is compared. Since we are proposing to use CF-CDS to replace CDS, other control message functionalities and signaling specifications remain the same. Therefore, we can compare the overhead difference of corresponding scheduling fields exclusively. In this particular shown scenario, the overall average number of neighbors is fixed, therefore the scheduling overhead of CDS is constant for all holdoff intervals. Alternatively, the scheduling overhead of CF-CDS increases if the holdoff interval increases. For most real world 802.16 deployment cases where the holdoff interval is usually small, CF-CDS actually generates less overhead than CDS. Furthermore, as long as the duration of control  $XmtOp$ , which is designed to be fixed and fairly large, is able to accommodate the size of the control message, the same portion of channel resource is consumed by the control sub-frame in the TDMA-based 802.16 airlink protocol. It is worth noting that theoretically, CF-CDS has higher node density upper bound than CDS. Let  $N$  denote the maximum number of nodes in any 2-hop neighborhood.

The holdoff interval of CF-CDS, which is also the number of bits of scheduling overhead for CF-CDS, only has to be set to equal to or larger than  $N$ . On the other hand, for CDS, the scheduling overhead is  $(8\text{bits} * N)$ , which is 8 times in size of CF-CDS scheduling overhead. Practically, the largest holdoff interval that the current CDS can support is constrained by the bits assigned to  $XmtHoldoffExponent::3\text{bits}$ . Based on (2), it requires a total of 256 bits fixed scheduling overhead for CF-CDS, a CDS scenario with equivalent scheduling overhead has 31 neighbors for every node. One can argue that this is still within an acceptable range.

## V. ENHANCED CF-CDS FOR DATA SUB-FRAME SCHEDULING

### A. Enhanced CF-CDS Algorithm

Given the periodic control signaling established by CDS, the actual scheduling for data transmissions are achieved by three-way handshakes (Request/Grant/Confirm) occurred between node pairs piggybacked in their control messages [1][3]. The Request part of the control message includes an availability map of the requestors data subframe, along with other necessary information for one or multiple requests (to different neighbors), such as ‘‘Demand Level (in minislots)’’, ‘‘Demand Persistence (in frames)’’. The Grant conveys the grantors decision regarding to reservation. The Requestor then completes the handshakes with the Confirm by repeating the reservation information of the Grant. Although the handshakes message formats are defined in the standard, detail resource allocation algorithms are left unstandardized to be vendor dependent. Baseline resource allocation algorithm could be fairly simple and straightforward. However, more efficient solution could be of high complexity in handling the following challenges:

Given all control messages are scheduled periodically (although not necessarily with the identical interval), it takes time for the control message handshakes to be completed. Especially, although nodes schedule control message transmissions periodically, the schedule interval is not deterministic. Furthermore, it may take more than one round of three-way handshakes to complete a reservation. Two major reasons are (i) the non-zero collision ratio shown in this paper, and (ii) conflicts incurred by concurrent reservations taking place among nodes, especially in a multihop environment. Therefore, such handshakes may incur substantial yet uncertain latency for data transmissions. This places challenge on setting the appropriate starting time of each data transmission in advance without knowing the actual control message handshake duration. Premature schedules will occur if handshakes complete after the scheduled starting time. On the other hand, large residual scheduling latency is undesirable if a node cannot start the transmission promptly after the completion of handshakes.

This dilemma on CDS data subframe scheduling leads us to extend the baseline CF-CDS into a full-blown scheduling protocol for data sub-frame scheduling that serves the needs of generic on-demand data transmissions. Again, based on its novel close-loop scheduling paradigm, enhanced CF-CDS successfully overcomes the challenges placed by the

forementioned 3-way handshaking approach. We believe that it serves as a promising alternative to fill the gap of data scheduling remained in the standard.

For the baseline CF-CDS, there is not problem to introduce an additional field of scheduling bitmap in the general MAC header. As long as the length of the scheduling cycle is limited to an acceptable length (i.e., the number of bytes incurred by the scheduling bitmap inside the MAC header is not substantial comparing with the remaining number of bytes to accommodate payload within the TDMA  $XmtOp$  interval), the communication overhead is acceptable. However, several limitations of the baseline version of CF-CDS have been observed. First, there is a hard limit on the maximum number of nodes within any neighborhood. If the number of neighbors is larger than the number of  $XmtOps$  within a scheduling interval, some nodes will never be able to scheduling any  $XmtOps$ . Second, a node only can, and always, reserves one  $XmtOp$  per scheduling cycle, which is evidently not optimal if the traffic distribution is not homogeneous within the network. Unfortunately, comparing with homogeneous traffic scenarios, heterogeneous traffic is ubiquitous since the traffic generated by nodes applications are varied, and the routing roles of nodes in a multiple topology may be varied largely. By simply adjusting the number of  $XmtOps$  per scheduling cycle, above problems cannot resolve satisfactorily. To solve aforementioned issues of baseline CF-CDS, we introduce four extensions as follows.

The first extension is to allow a node to reserve more than one hello slots within a single scheduling cycle. This can be easily achieved by maintaining multiple tuples of scheduling parameters  $TB$ ,  $LB$ ,  $myXmtOp$ , and  $ifTx$ . Different sets of parameters operate independently using the same algorithm described in Figure 6, with the only additional constraint that each has a unique  $myXmtOp$ . This extension can be taken as that, instead of a single scheduler, there are multiple baseline CF-CDS schedulers cooperating at each node. This extension enables a node to schedule a configurable number of  $XmtOps$  within the scheduling cycle, which relaxes the constraint that a node can only transmit once per scheduling cycle. From a slightly different angle, it has similar effect of allowing non-identical  $XmtHoldoffExponents$  among nodes. Interest readers may find potential QoS usage of this aspect in [9].

The second step of the extension is to develop feedback controller that advise the node how many  $XmtOps$  it should schedule for a specific scheduling cycle. This is to ensure that the resources of  $XmtOps$  are efficiently allocated among nodes in the neighborhood. Evidently, if a node does not have many data packet to send during certain period of time, it should not reserve a large number of  $XmtOps$  during that time. On the other hand, if increasing queue backlogged has been observed, a node should be able to schedule more  $XmtOps$  promptly. In other words, the controller should manage the number of  $myXmtOps$  dynamically on-demand. With this extension, a node does not have to reserve any  $XmtOp$  if it does not have any traffic, and it can reserve/release appropriate amount of  $XmtOps$  based on the temporal variations of its queue depth. However, its worth noting that if the controller is designed to be over-sensitive to the variation of queue depth, frequent rescheduling may introduce unnecessary collisions during the

convergence period of scheduling. Instead of using transient queue depth, Simple Moving Average (SMA) or Exponential Moving Average (EMA) with a configurable sampling window size ( $L_{MA}$ ) can be used as the input of the controller. Given the measured average queue depth,  $N_{QueueDepth}$ , the number of  $XmtOps$  should be reserved,  $N_{XmtOp}$  is calculated as the following

$$N_{XmtOp} = \begin{cases} 2^{ceil(\frac{N_{QueueDepth}}{T_{QueueDepth}})} & 2^{ceil(\frac{N_{QueueDepth}}{T_{QueueDepth}})} < n_{XmtOp}^{Max} \\ n_{XmtOp}^{Max} & \text{o.w.} \end{cases} \quad (3)$$

where  $T_{QueueDepth}$  is a configurable threshold corresponding to the number of  $XmtOps$  per scheduling cycle. And  $n_{XmtOp}^{Max}$  is the maximum number of  $XmtOps$  allowable to single node, which is configurable and may not necessarily be identical among nodes. In (3), we choose to exponentially increase the scheduling “valve” (i.e., number of  $XmtOps$ ) according to the average queue depth. Other approaches, such as linear increase, or even TCP-like congestion control mechanism, may be further evaluated.

A desirable outcome of this extension is that it relaxes the topology constraint previously imposed on the baseline CF-CDS. In other words, the minimum number of  $XmtOps$  per scheduling cycle may not have to be equal or larger than the maximum number of nodes within a 2-hop scheduling neighborhood, i.e. the maximum node density. A smaller scheduling cycle will still be able to serve a dense network since nodes will release their reserved slots once their traffic demands have been satisfied. This relaxation limits the scheduling overhead of CF-CDS and saves the headache of configuring appropriate scheduling cycle for different network deployments. A reasonable 32-bits bitmaps, which is small overhead in the MAC header, may be sufficient for all-purpose use.

The third extension is to alleviate the scheduling convergence introduced by on-demand scheduling and small scheduling cycle wrt. large number of contending nodes in the neighborhood. We add one more mechanism to alleviate the scheduling convergence by introduce probabilistic scheduling when a node picks an  $XmtOp$  (in Line 18 and 21 of Figure 6). Given such randomly picked  $XmtOp$ , a node will actually use it with a probability,  $P_{sch}$ , which is the following

$$P_{sch} = \frac{n_{idle}}{N_{cycle}} \quad (4)$$

where  $n_{idle}$  is the number of idle  $XmtOp$  currently available based on a nodes LBs.

The last major extension, which is very unique to CF-CDS algorithm, is to introduce ARQ mechanism so that any collided transmission during the convergence can be retransmitted instead of lost. ARQ itself is not unique for reliable wireless MAC services. However, the way ARQ is inherently and efficiently integrated in CF-CDS is unique. First, given CF-CDSs proactive close-loop monitoring of transmissions within the neighborhood, ARQ can simply tag along with the same scheduling overhead. Second, CF-CDS supports both unicast and broadcast ARQ. ARQ feature can be easily added to CF-CDS at Line 37-39 in Figure 6. As described in the algorithm,

after a complete scheduling cycle of a specific  $myXmtOp$ , node can automatically identify whether any collision has occurred for the previous transmission, at the discretion of the detail feedback of each individual neighbor. Therefore, if collision is detected (either from a specific unicast neighbor or a set of broadcast neighbors), the node can retransmit the previous payload (not the scheduling header) in any other scheduled or rescheduled  $myXmtOp$ . Note that we do not claim zero additional overhead for ARQ, since the next-hop destination may have to explicitly acknowledge the sender in some case even though it does not have any traffic need itself in the enhanced CF-CDS. Instead of enforcing the classical strict ARQ, a design tradeoff is to enforce a quasi-ARQ where third-party (non-next-hop-destination) acknowledgements are leveraged. For example, the number of third-party acknowledgements can be a useful indicator for both successful unicast and broadcast transmissions. Of course, there is a possibility that a successful transmission by quasi-ARQ is in fact not successful at the intended receiver(s). In this paper, we do not use quasi-ARQ.

In summary, with above four extensions, the enhanced CF-CDS has the capability of support reliable data transmissions in distributed TDMA fashion and in generic multi-hop topology. The overhead of enhanced CF-CDS remains a constant when the network size increases, which is remarkable in terms of scalability.

## B. Simulation Results

We compare the data scheduling performance between the standards 3-way handshakes (we call it “CDS” in the following just for convenience) and the enhanced CF-CDS under various traffic loads. To isolate other factors, we test two scenarios, with simple 1-hop topology of 26 nodes and 51 nodes, respectively. In other words, all nodes are within the 1-hop neighborhood of each other. For both scenarios, one node is the receiver, and the rest of nodes generate application traffic to the receiver.

For CF-CDS, in addition to simulation parameters in Table I, we assume fixed length of  $XmtOp$  of 15 minislots in the data sub-frame. Each data sub-frame consists of 17  $XmtOps$ . Each scheduling cycle consists of 32  $XmtOps$ , which contributes to a merely 4 bytes scheduling communications overhead per transmission. We use SMA algorithm to calculate the average queue depth, with a sampling window size,  $L_{MA}$ , of 64  $XmtOps$ .  $T_{QueueDepth}$  is set to a large number of 100 packets to minimize unnecessary convergence.  $n_{XmtOp}^{Max}$  is set to 32, i.e., the number of  $XmtOps$  in a scheduling cycle. For CDS, a similar SMA algorithm is implemented, with other configurations matching CF-CDSs. In Figure 9, identical CBR and Poisson traffic flows with fixed packet length of 512 bytes and varied packet inter-arrival times at all senders are tested. The starting time of traffic flows are randomized within a limited time window. However, all traffic flows coexist for enough long period of time to generate aggregated traffic load, which is varied by adjusting the packet interval. The smallest packet intervals are 0.012s and 0.006s for Figure 9(a) and (b), respectively, which generate aggregated traffic load close to 100% of TDMA MAC capacity (number of  $XmtOps$  per second) in both scenarios. Both CDS and CF-CDS maintain

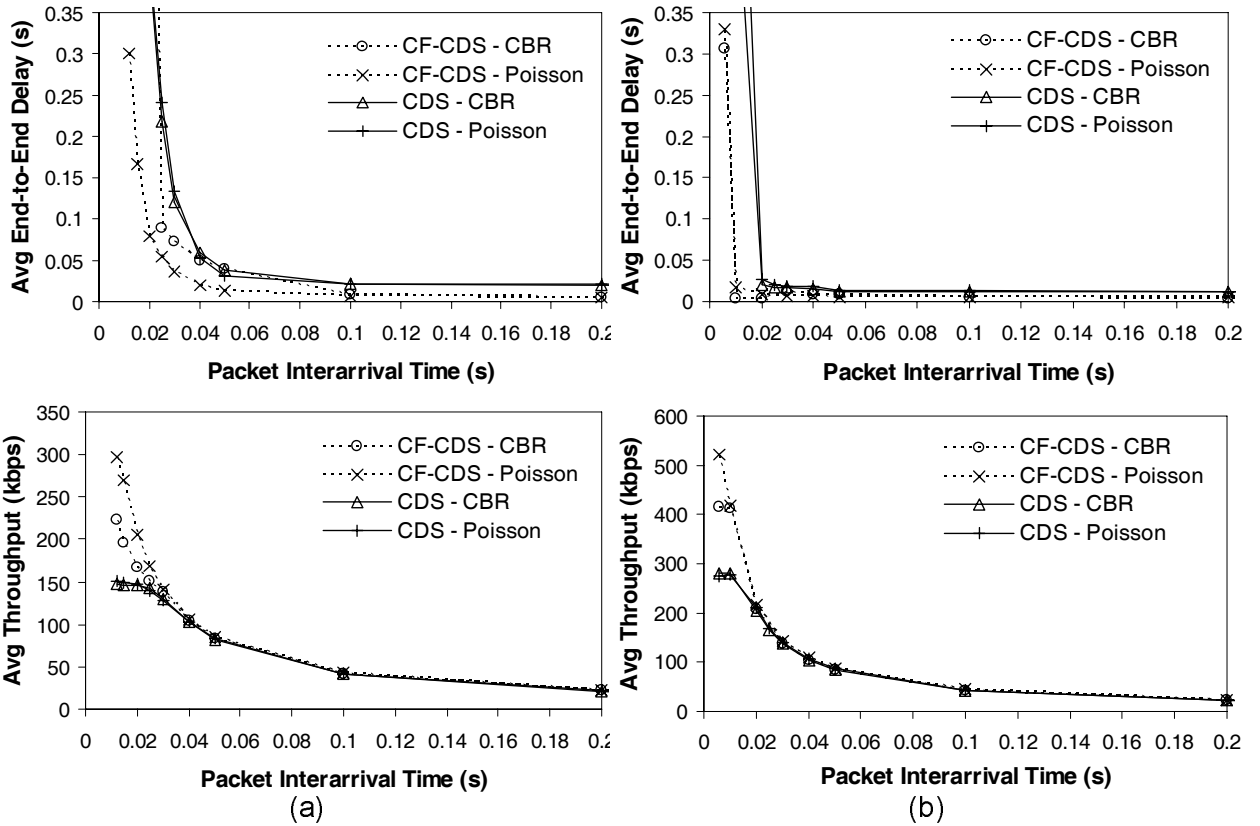


Fig. 9. Average application end-to-end delay (a) 51 nodes, (b) 26 nodes

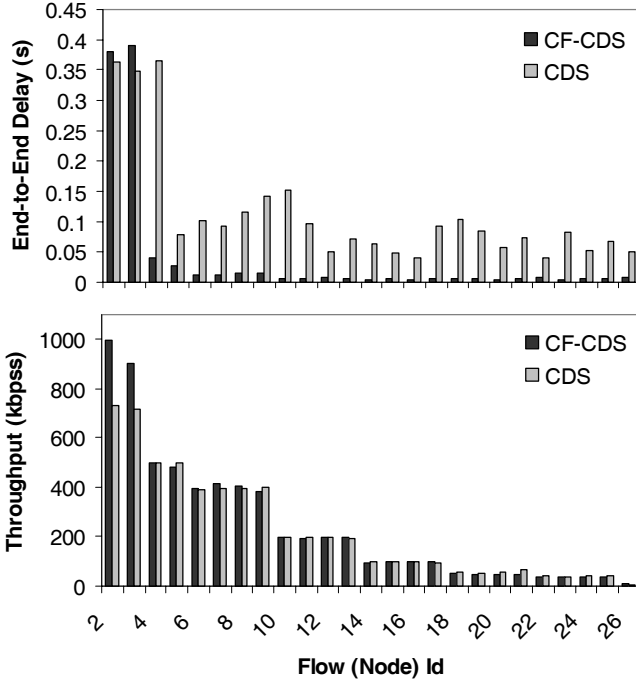


Fig. 10. Per-flow performance with heterogeneous traffic

small end-to-end delay under low traffic load, with CF-CDS slightly smaller. Larger delay of CDS is due to the 3-way handshakes. However, CF-CDS achieves significant higher throughput and smaller delay than that of CDS under moderate to high traffic load. Figure 10 shows the per-traffic-flow

performance for moderate heterogeneous traffic. Among 25 Poisson traffic flows, two are of 1 Mbps, two are of 500 kbps, four are of 400 kbps, four are of 200 kbps, four are of 100 kbps, four are of 50 kbps, four are of 40 kbps, and one is of 8 kbps. For CF-CDS, most of traffic flows experience end-to-end delay less than 20ms while 4 high-data-rate flows experience large delay of 350ms. On the other hand, CDS experiences much larger delay for all low-data-rate flows. Moreover, CDS shows notable throughput degradation for high-data-rate flows.

## VI. RELATED WORKS

As relatively new standard, IEEE 802.16 has been studied much less than access technologies such as IEEE 802.11. Eklund *et al.* presented a system level overview of 802.16 standards family in [10]. Redana and Lott modeled and compared the control message overhead between centralized and distributed scheduling mechanisms in [11]. From a different angle, Cao *et al.* proposed a theoretic model to compute the schedule interval of 802.16 coordinated distributed scheduling in [9]. With the algorithm to grant data requests left open in the standard, the schedule interval is an important common performance metric that reflects the scheduling latency of coordinated distributed scheduling. Both general formulation and practical computable approximation under the assumption of geometric distribution of scheduling attempts after *EarliestSubsequentXmtTime* are presented. The model is validated though NS-2 simulation, which matches well with our QualNet results.

Multi-hop scheduling is one of the fundamental problems of ad hoc networks because the capacity of ad hoc networks

degrades substantially when the network size increases [4]. Although distributed scheduling algorithms have been studied for other access technologies as upper layer components, slot scheduling for the medium access is a signature problem for TDM-based MAC problem. For a peer-to-peer TDMA system in which there is no logic difference between uplink and downlink, there are a large amount of research work resort to additional mechanisms from PHY, such as introducing CDMA overlay, or emulating full-duplex time slot consisting of paired or more mini slots for signaling back and force, or relying on the aggressive multi-channel network architecture, etc. However, there are a few algorithms that are able to achieve distributed TDMA slot scheduling without significant modifications of simple TDMA radios. We give two representatives as follows.

One such algorithm is SEEDEX, developed by Rozovsky and Kumar [12]. SEEDEXs idea of using common seed construction rule and letting nodes independently generate predictable pseudo random schedules was innovative and very similar to current CDS of 802.16. However, SEEDEX does not intend to create collision-free scheduling, instead it only uses seed information aggressively to minimize the number of active contending 2-hop neighbors in a Slotted-Aloha access method. As a result SEEDEX is expected to have performance improvement over Slotted-Aloha. However, it has been observed that the performance of SEEDEX is sensitive to certain parameters.

Another class of algorithms includes NAMA, LAMA, PAMA and HAMA, developed by Bao and Garcia-Luna-Aceves [13]. Among all four schemes, only NAMA does not require additional CDMA overlay. These algorithms adopt the same idea of seed construction. However, they use the available neighbor information conservatively to avoid concurrent transmissions within 2-hop neighborhood. It is worth noting that, like 802.16 CDS, these algorithms are vulnerable to non-quasi-interference channel as well. Therefore true "collision-free" scheduling is not achieved. Note that both SEEDEX and NAMA fall into the same paradigm of 802.11 CDS, that letting nodes schedule slots independently with shared random seeds and predictable pseudo random generator. This type of coordination is not enough to prevent collisions due to accumulative interference in realistic non-quasi-interference environments.

Finally, the proposed CF-CDS algorithm and its node coordination paradigm were originally designed for distributed link scheduling in mobile MIMO ad hoc networks [14], where the traditional quasi-interference modeling with the omni-antenna does not hold.

## VII. CONCLUSION

In this paper, we proposed baseline and enhanced CF-CDS for 802.16 Mesh control sub-frame and data sub-frame scheduling, respectively. Through extensive simulations, we show that (i) the baseline CF-CDS outperforms existing CDS for periodic control scheduling by truly achieving collision-free scheduling with spatial reuse, and (ii) the enhanced CF-CDS outperforms current three-way handshakes data scheduling in both throughput and delay.

## REFERENCES

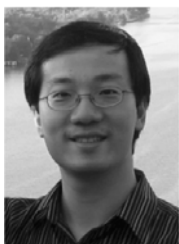
- [1] IEEE Standard 802.16-2004, "IEEE Standard for Local and Metropolitan Area Networks – Part 16: Air Interface for Fixed Broadband Wireless Access Systems," Oct. 2004.
- [2] IEEE Standard 802.16.2-2004, "IEEE Recommended Practice for Local and Metropolitan Area Networks Coexistence of Fixed Broadband Wireless Access Systems," 2004.
- [3] IEEE Std 802.16-2004/Cor1, "Corrigendum to IEEE Standard for Local and Metropolitan Area Networks - Part 16: Air Interface for Fixed Broadband Wireless Access Systems," Draft 5, 2005-09-12.
- [4] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inform. Theory*, vol. 46, no. 2, pp. 388-404, 2000.
- [5] M. Nohara, "Ad hoc meeting report: mobile multihop relay networking in IEEE 802.16," document IEEE 802.16-05/51, 21 July 2005.
- [6] M. Asa, D. T. Chen, and N. Natarajan, "Concepts for 802.16-based mobile multihop relay networking," document IEEE C802.16-05/15, 19 July 2005.
- [7] D. Beyer, N. van Waes, and C. Eklund, "Tutorial: 802.16 AC layer mesh extensions overviews," IEEE 802.16 (document S802.16a-02/30), St. Louis, 11 Mar., 2002.
- [8] *QualNet User's Manual*, version 3.8, Scalable Network Technologies, Inc. 2005, available at <http://www.scalable-networks.com/>.
- [9] M. Cao, W. Ma, Q. Zhang, X. Wang, and W. Zhu, "Modeling and performance analysis of the distributed scheduler in IEEE 802.16 mesh mode," in *Proc. 6th ACM Intl Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc05)*, Urbana-Champaign, IL, 2005, pp. 78-89.
- [10] C. Eklund, R. Marks, K. L. Stanwood, and S. Wang, "IEEE standard 802.16: a technical overview of the *WirelessMAN<sup>TM</sup>* air interface for broadband wireless access," *IEEE Commun. Mag.*, pp. 98-107, June 2002.
- [11] S. Redana and M. Lott, "Performance analysis of IEEE 802.16a in mesh operation mode," in *Proc. 13th IST SUMMIT*, Lyon, France, June, 2004.
- [12] R. Rozovsky and P. R. Kumar, "SEEDEX: a MAC protocol for ad hoc networks," in *Proc. ACM Mobihoc01*, Oct. 2001.
- [13] L. Bao and J. J. Garcia-Luna-Aceves, "Distributed Channel Access Scheduling for Ad Hoc Networks," book chapter in *Algorithms and Protocols for Wireless and Mobile Networks*, A. Boukerche (Ed.), CRC/Hall Publisher, 2004.
- [14] A. Gummalla and G. Nallamothu, "Link scheduling media access protocol for mobile MIMO ad hoc networks," in *Proc. SPIE Symposium on Defense and Security*, Apr. 2006.
- [15] Standard ECMA-368, "High Rate Ultra Wideband PHY and MAC Standard," Dec. 2005, available at <http://www.ecma-international.org/publications/standards/Ecma-368.htm>



**Hua Zhu** received his Ph.D. in Electrical Engineering from the University of Texas at Dallas where he was a member of the Center for Advanced Telecommunications Systems and Services. He is currently with San Diego Research Center.

Dr. Zhu's research interests include algorithmic and protocol design, modeling and simulation for wireless communications systems and networks. He has worked extensively on various wireless access technologies, including both commercial IEEE 802.11/WiFi, 802.16/WiMax, and tactical communication systems. Dr. Hua Zhu is the recipient of Texas Telecommunications Engineering Consortium (TxTEC) Scholarship for his research in wireless multimedia streaming. He is the Co-PI or key personnel of several US Government funded projects on ad hoc and sensor networks, and robotic systems.

Dr. Zhu has served on the technical program committees of several international conferences, including IEEE ICC'09, IEEE VTC'08 Spring and Fall, IEEE ICCCN'08, ChinaCom'09, TridentCom'09, AccessNets'06-'08, CIT'06-'07, IEEE ISWPC'07-08. He also served on the organizing committees of the Workshop on Optical Networking Technologies for Global SAN Solutions (ONSAN'03), the Int'l Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine'04), and the Int'l Conference on Broadband Networks (BroadNets'04).



**YatKwan Tang** received the B.Eng. and M.Phil. degrees in Computer Science from Hong Kong University of Science and Technology, Hong Kong in 1999 and 2002, respectively. He also received the M.S. degree in Electrical Engineering from University of Texas, at Dallas, U.S., in 2004.

Currently, he is working as Network Research Engineer at San Diego Research Center. His research interests include admission control and handoff management in WCDMA systems, power control in sensor networks, and media access control for wireless

ad hoc networks.



**Imrich Chlamtac** holds a Ph.D. in Computer Science from the University of Minnesota (1979). He received his B.Sci. and M.Sci. degrees in mathematics with Highest Distinction from Tel Aviv University (1977). Dr. Chlamtac is the President of CREATE-NET, a European research consortium. Dr. Chlamtac was the Distinguished Chair in Telecommunications endowed professor, the Director of CATSS, and Associate Provost for Research at the University of Texas at Dallas. Prior to joining UTD, Dr. Chlamtac was on faculty at Technion - the Israel

Institute of Technology, the University of Massachusetts at Amherst and the Photonics Center. Dr. Chlamtac holds several honorary appointments including the Bruno Kessler Honorary Professor, University of Trento, Italy, the Sackler Professorship at Tel Aviv University, Honorary Professorship at the Beijing University of Posts and Telecommunications, the "University Professorship" at the Budapest University of Technology and Economics (BUTE), Hungary, a Doctor of the Hungarian Academy of Sciences and Honorary Membership of the BUTE Senate. Dr. Chlamtac is the co-founder and past President of Consip, the first network emulator company, and of BCN, one of the largest system integrator companies in central Europe.

In 1993 Dr. Chlamtac was elected Fellow of the IEEE for his work on ad-hoc access protocols and in 1997 he became a Fellow of the ACM for introducing the concept of lightpaths, the foundation for today's optical WDM networks. In 1994 he received the Fulbright Scholarship. He is the 2001 recipient of the ACM Award for Outstanding Contributions to Research on Mobility and the 2002 recipient of the IEEE Award for Outstanding Technical Contributions to Wireless Personal Communications. Dr. Chlamtac is a winner of the New Talents in Simulations Award from the Society of Computer Simulation for the concept of network emulators (1980) and the recipient of multiple ACM and an SPIE best paper awards. He has lectured worldwide as IEEE Distinguished Lecturer (1993 and 2000-2001), and was the plenary and keynote speaker at leading conferences.

Dr. Chlamtac is the founding Editor in Chief of the ACM/SPRINGER WIRELESS NETWORKS (WINET), the ACM/SPRINGER JOURNAL ON SPECIAL TOPICS IN MOBILE NETWORKS AND APPLICATIONS (MONET). He served as EIC of the SPIE OPTICAL MAGAZINE and was on the editorial boards and advisory boards of IEEE TRANSACTIONS ON COMMUNICATIONS, COMPUTER NETWORKS AND ISDN SYSTEMS, HIGH SPEED NETWORKS JOURNAL, TELECOMMUNICATION SYSTEMS, the PHOTONIC NETWORK COMMUNICATIONS JOURNAL and the JOURNAL OF GRAPH ALGORITHMS AND APPLICATIONS. Dr. Chlamtac was the General Chair of leading ACM and IEEE conferences and is also the founder and Steering Committee Chair of ACM/IEEE MobiCom, the world premier conference on mobility. Dr. Chlamtac is the founder and past Chairman of ACM SigMobile, the Special Interest Group on Mobile Computing and Networking.