Proceedings of APCC2008 copyright © 2008 IEICE 08 SB 0083

# An Enhanced Election-Based Transmission Timing Mechanism in IEEE 802.16 Mesh Networks

Bih-Hwang Lee
lee@ccg.ee.ntust.edu.tw

Chun-Ming Chen
issu@ccg.ee.ntust.edu.tw

Department of Electrical Engineering
National Taiwan University of Science and Technology

*Abstract-* **IEEE 802.16, also known as the worldwide interoperability for microwave access (WiMAX), defines point-to-multipoint (PMP) and mesh modes to support wireless medium access. The mesh mode provides an easy and cheap way to construct the last-mile connection. The multi-hop environment with spatial reuse can use the radio resource more efficiently. IEEE 802.16 defines three kinds of scheduling algorithms without specifying the detail. In this paper, we propose a new enhanced election-based transmission timing mechanism to solve the problem currently on the coordinated distributed scheduling algorithm. According to the simulation results by using the network simulator ns2, the new mechanism can prevent the unexpected collisions of MSH-DSCH messages to gain better performance on time-sensitive traffic.**

*Keywords:* IEEE 802.16, WiMAX, wireless medium access, scheduling algorithms

## I. INTRODUCTION

The demand of broadband access is highly increased by the rapid growth of new services, such as voice over IP (VoIP), video on demand (VoD), online gaming, and peer-to-peer. It is a challenge to provide broadband access everywhere, hence wireless access technologies have become attractive solutions. Compared with the wired technologies such as HFC (hybrid fiber/coaxial) and DSL (digital subscriber line), the broadband wireless access (BWA) can be deployed easily with low cost regardless of the terrain layouts, and it is simple to add users by providing additional radio interface and capable link conditions[1][2]. IEEE 802.16, also known as the worldwide interoperability for microwave access (WiMAX), defines point-to-multipoint (PMP) and mesh modes to support wireless medium access[3]. In the PMP mode, the wireless link operates with a central base station (BS) to serve a group of subscriber stations (SSs) within the same antenna sector by broadcasting. All SSs receive the same transmission signal from BS within a given frequency channel, while BS coordinates all the transmissions from SSs to BS. On the other hand, SSs can communicate with each other directly without BS involved in the mesh mode.

The IEEE 802.16 standard defines three kinds of scheduling algorithms for the mesh mode: the centralized scheduling, the coordinated distributed scheduling, and the uncoordinated distributed scheduling. In the centralized scheduling, BS determines the flow assignments on the channel by the requests of SSs; SSs then use the information from BS to realize how to share the channel in different time slots. The connection setup time is extremely long, because all control and data packets have to go through BS from source to destination; therefore the centralized scheduling is not suitable to occasional traffics. In both coordinated and uncoordinated distributed schedulings, the IEEE 802.16 standard defines a three-way handshake mechanism to allocate the channel resources among SSs. The coordinated distributed scheduling uses the control subframe to transmit its own scheduling. By contrast, the uncoordinated distributed scheduling is established by direct requests and grants in the data subframe without colliding with data scheduled by the other schedulings.

To compete the transmission opportunities in the control subframe, the IEEE 802.16 standard proposes a pseudo-random election algorithm based on the scheduling information in the two-hop neighborhood. The coordinated distributed scheduling transmits its signaling packet by this election algorithm. However, IEEE 802.16 standard does not specify the detail of the competing method. Ref. [4] proposes an extension of original election algorithm to solve the problem that information from neighbors which are 2 hops far is not correct. But the extension does not really fix the situation. This paper proposed a modification on the extension to overcome the problem and examine its performance on ns2.

The rest of this paper is organized as follows. Sec. II describes the overview of the coordinated distributed scheduling algorithm in the IEEE 802.16 mesh mode. Sec. III states the problem on current election-based transmission timing (EBTT) mechanism and provides our enhancement on EBTT mechanism. We show our simulation results in Section IV. Finally conclusions are made in Section V.

## II. THE COORDINATED DISTRIBUTED SCHEDULING ALGORITHM

In the IEEE 802.16 scheduling algorithms, the control messages and data packets are allocated in different timeslots as Fig. 1. The descriptions of the different control messages and the related scheduling parameters are shown in Table I. The control subframe provides two functions: one is for network construction and maintenance, known as network control subframe, and the other is for resource scheduling, known as schedule

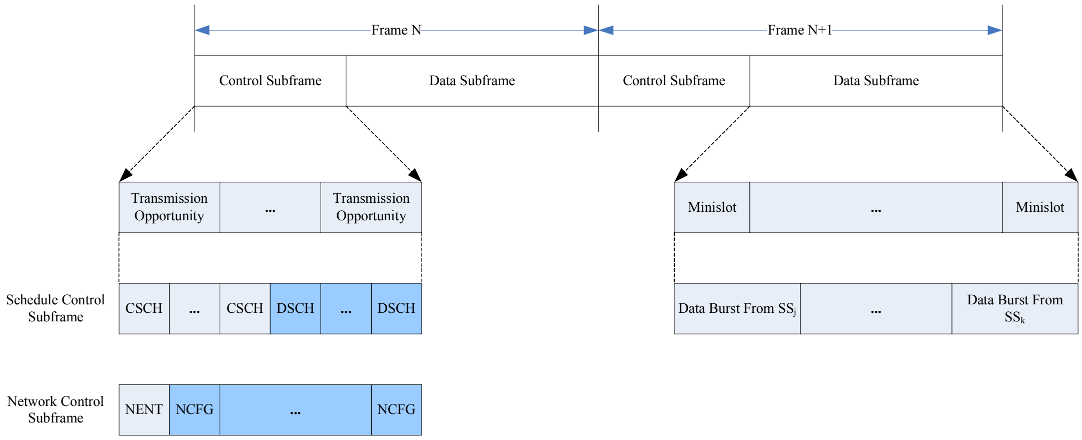


Figure 1. The IEEE 802.16 frame structure for the mesh mode.

control subframe. In network control subframe, the MSH-NENT message is used for the new node to send network entry request, while the MSH-NCFG message obtains network parameters. In schedule control subframe, the MSH-CSCH message stores the centralized scheduling information, while the MSH-DSCH message stores the coordinated distributed scheduling information. Frames with the network control subframe happen periodically every [*Scheduling_Frames* + 1] frames; and all the other frames contain the schedule control subframes. Every control subframe consists of *MSH-CTRL-LEN* transmission opportunities. Each transmission opportunity has seven OFDM symbols and carries one control message. The value of *MSH-DSCH-NUM* defines how many MSH-DSCH messages in one schedule control subframe. Therefore, the length of the MSH-CSCH parts is equal to *MSH-CTRL-LEN – MSH-DSCH-NUM.*

The coordinated distributed scheduling algorithm employs a three way handshaking to setup the transmission as shown in Fig. 2. The requester first sends a request with or without availability information of possible minislots assignment via its MSH-DSCH message to the granter. The granter then verifies if the request can be fulfilled. If so, it feedbacks a grant message with the assigned minislot information to the requester.

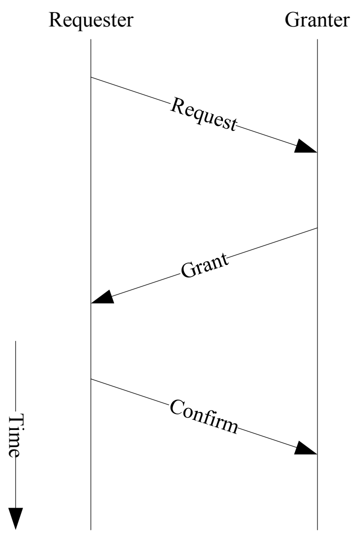


Figure 2. Three way handshaking procedure.

The requester sends confirm message back to the granter once it accepts the grant message.

TABLE I
IEEE 802.16 MESH MODE CONTROL MESSAGES AND PARAMETERS

| Message/Parameter | Description |
|---|---|
| MSH-NENT | Mesh network entry message |
| MSH-NCFG | Mesh network configuration message |
| MSH-DSCH | Mesh distributed schedule message |
| MSH-CSCF | Mesh centralized schedule configuration message |
| MSH-CSCH | Mesh centralized schedule message |
| MSH-CTRL-LEN | Control subframe length |
| MSH-DSCH-NUM | Number of MSH-DSCH opportunities in schedule control subframe |
| Scheduling_Frames | Defines how many frames have a schedule control subframe between two frames with network control subframe |
| NextXmtMx | A configuration parameter used to calculate NextXmtTimeInterval |
| NextXmtTime | Dictates the next MSH-NCFG or MSH-DSCH message transmission opportunity |
| NextXmtTimeInterval | Dictates the next MSH-NCFG or MSH-DSCH message eligibility interval for neighbors |
| XmtHoldoffTime | Dictates the number of MSH-NCFG. MSH-DSCH transmission opportunities an SS can not transmit such messages after NextXmtTime |
| XmtHoldoffExponent | A configuration parameter used to calculate XmtHoldoffTime |
| TempNextXmtTime | The temporary next transmission opportunity used in election-based transmission timing mechanism to compete with competing neighbors |
| EarliestSubsequentXmtTime | Estimates the earliest transmission opportunity an SS is eligible to transmit MSH-NCFG or MSH-DSCH message after NextXmtTime |

## III. THE ENHANCED ELECTION-BASED TRANSMISSION TIMING (EEBTT) MECHANISM

### *A. IEEE 802.16 Election-Based Transmission Timing (EBTT) Mechanism*

IEEE 802.16 EBTT mechanism is used to schedule the coordinated MSH-DSCH and MSH-NCFG messages in control subframe without explicit schedule negotiations [5]. This mechanism is supposed to be collision-free within each node's extended neighborhood. Moreover, it is completely distributed and needs no central control from BS. Besides, it is also fair and robust due to the pseudo-random nature. To avoid the collision of the MSH-DSCH messages in the schedule control subframe, every node needs to broadcast the information of its next MSH-DSCH transmission time, called *NextXmtTime* in the following, to its neighbors. *NextXmtTime* of each node is described by two parameters, *NextXmtMx* and *XmtHoldoffExponent*, in IEEE 802.16. These two parameters are embedded in every MSH-DSCH message to form an eligibility interval, *NextXmtTimeInterval*, where the exact *NextXmtTime* is located. *NextXmtTime* can be expressed as: $2^{XmtHoldoffExponent} \cdot NextXmtMx < NextXmtTime \le 2^{XmtHoldoffExponent} \cdot (NextXmtMx + 1)$. The length of *NextXmtTimeInterval* is $2^{XmtHoldoffExponent}$. Besides broadcasting the node's own parameters via MSH-DSCH, it also broadcasts its all one-hop neighbors' parameters. By doing so, every node is able to calculate *NextXmtTimeInterval* of the nodes in its two-hop neighborhood. After the successful transmission of the MSH-DSCH message, a node must wait at least *XmtHoldoffTime* transmission opportunities to start its election procedure again. *XmtHoldoffTime* can be expressed as: $XmtHoldoff\ Time = 2^{XmtHoldoff\ Exponent\ +4}$.

Every node calculates its *NextXmtTime* during the current transmission time using the distributed election algorithm defined in the standard. A node chooses the first transmission opportunity after the holdoff time as the temporary next transmission time, *TempNextXmtTime*, and competes at this opportunity with all possible competing nodes in the two-hop neighborhood as shown in Fig. 3. $L_k$ denotes the length of *NextTmxTimeInterval* so that $XmtHoldoffTime_k = 16L_k$. There are three types of competing nodes: (1) node whose *NextTmxTimeInterval* includes the *TempNextXmtTime* (Node B); (2) node whose *EarliestSubsequentXmtTime* is the same as or before the *TempNextXmtTime*, while $EarliestSubsequentXmtTime = NextTmxTime + XmtHoldoffTime$ (Node C); and (3) node whose scheduling information is unknown (Node D). Once the competing nodes are found, a pseudo-random election function is used to decide the winner. If the node wins, it sets the *TempNextXmtTime* for its *NextXmtTime* and computes the corresponding *NextXmtMx* by its *XmtHoldoffExponent*. Then the node broadcasts the information to the neighbors by the MSH-DSCH message. If the nodes looses, it chooses the next transmission opportunity and repeats the competing procedures until it wins.

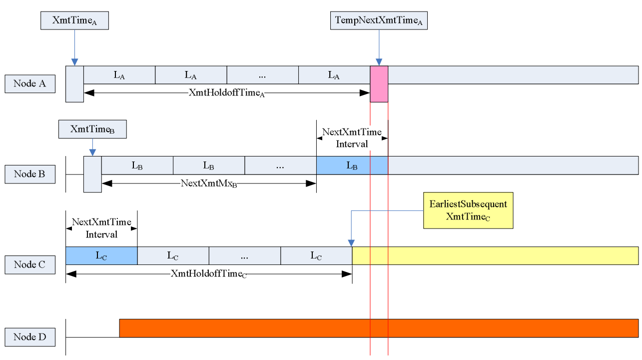


Figure 3. Competing Node Decision.

*B. Problem Statements of IEEE 802.16 EBTT Mechanism*

Since the calculation of *NextXmtTimeInterval* is based on *NextXmtMx*, *XmtHoldoffExponent* and the transmission opportunity when the MSH-DSCH message is received, the scheduling information of the neighbors that are 2 hops far will be miscalculated as shown in Fig. 4. Node K is the neighbor of node J. Node I is the neighbor of node J; and node K is 2 hops far from node I. Let $XmtTime_N$ denotes the MSH-DSCH message transmission time of node N. Once node I receives MSH-DSCH message from Node J, it will calculate *NextXmtTimeInterval* of Node K based on $XmtTime_J$ instead of $XmtTime_K$. Because $XmtTime_K$ is different from $XmtTime_J$, *NextXmtTime* of Node K has the chance to be miscalculated to cause the unexpected collision. Since there is no way for Node I to know the actual $XmtTime_K$, the scheduling information of Node K should be modified before Node J broadcasting. Ref. [4] proposes an extension to IEEE 802.16 EBTT (EEBTT) which modifies *NextXmtMx* using (1). However, this extension does not entirely solve the problem. Fig. 5 shows the possible problem that EEBTT may experience. Node I, J and K have the same relationship as the previous scenario. Since *NextXmtTime* is located in *NextXmtTimeInterval*, it might occur at the very early or late opportunity of *NextXmtTimeInterval*. Case (a) shows the situation that *NextXmtTime* occurs at the first transmission opportunity of *NextXmtTimeInterval*. By contrast, case (b) represents the opposite situation that *NextXmtTime* occurs at the last transmission opportunity of *NextXmtTimeInterval*. Apparently, in case (a), Node I still gets a wrong *NextXmtTimeInterval* estimation even using EEBTT.

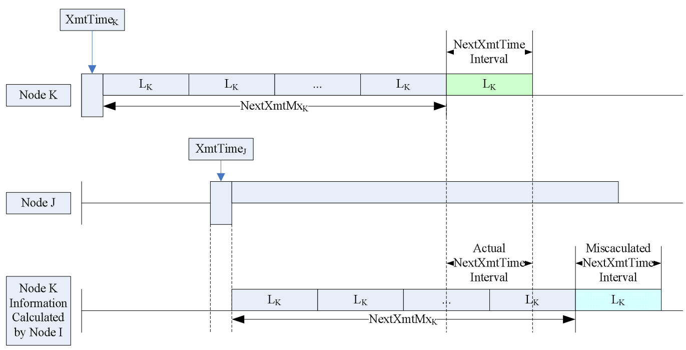


Figure 4. Mismatch of the *NextXmtTimeInterval*.

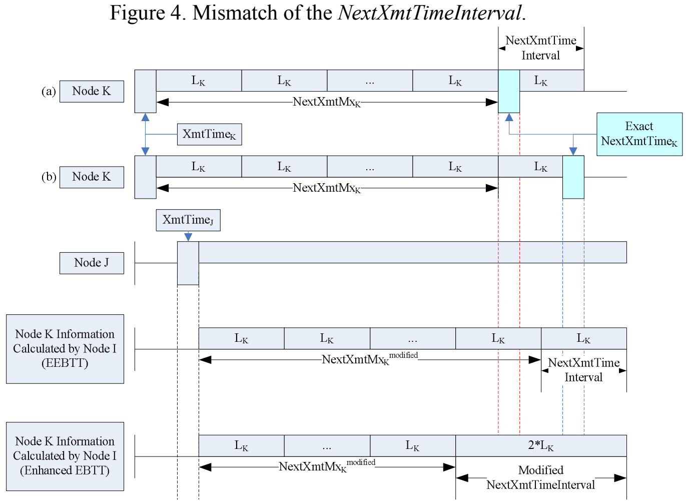


Figure 5. Comparison of EEBTT and Enhanced EBTT.

$$NextXmtMx_K^{\text{modified}} = floor\left( \frac{NextXmtTime_K - floor\left( \frac{XmtTime_J}{2^{XmtHoldoffExponent_K}} \right) * 2^{XmtHoldoffExponent_K} - 1}{2^{XmtHoldoffExponent_K}} \right) \quad (1)$$

$$NextXmtMx_K^{\text{modified}} = floor\left( \frac{NextXmtTime_K - XmtTime_J - 1}{2^{XmtHoldoffExponent_K}} \right) \quad (2)$$

$$2^{XmtHoldoffExponent} \cdot NextXmtMx < NextXmtTime \le 2^{XmtHoldoffExponent} \cdot \left( NextXmtMx + 2 \right) \quad (3)$$

### C. The enhanced EBTT mechanism

As the description in the previous section, both original IEEE 802.16 EBTT and EEBTT can not completely avoid the possible collision with the MSH-DSCH messages from neighbors which are 2 hops far. We make two modifications on the original IEEE 802.16 EBTT mechanism to solve the problem. First at all, *NextXmtMx* has to be modified to compensate the difference between Node K and Node J as (2).

The floor function used to modify *NextXmtMx* makes the modified value of *NextXmtMx* be early or equal to the correct one. Therefore, the original length of *NextXmtTimeInterval* is not enough to cover all the *NextXmtTime* possibilities from neighbors that are 2 hops far. To overcome this limitation, we propose (3) by doubling *NextXmtTimeInterval*. As shown in Fig. 5, the combination of these two modifications ensures the modified *NextXmtTimeInterval* covering all the possibilities. Since we can get the hop distance information in the MSH-NCFG messages to justify whether the neighbor is 2 hops far or not, the modification only applies to the *NextXmtTimeInterval* calculations of the neighbors that are 2 hops far, and the rest nodes keep unchanged. In doing so, we can alleviate the additional delay caused by the increased competition probability.

## IV. Simulation Results

The network simulator ns2 [6] is used to realize the performance of our enhancements on EBTT. Thus we have developed a WiMAX mesh module for ns2 based on the IEEE 802.16 standard. The module is modified from the open source code of [7]. Four different EBTT mechanisms are compared in our simulation. The first one, termed as unaware, is the original IEEE 802.16 EBTT mechanism, and it treats all neighbors which are 2 hops far as unknown nodes. In opposite to the first one, all exact *NextXmtTime* of the neighbors in the two-hop neighborhood are known in the second mechanism, termed as aware. It should be the best mechanism compared with the others since it has the least competition probability. The third one is EEBTT; and the last one is our proposed enhanced EBTT. All scenarios are based on an equilateral grid network consists of 36 nodes shown as Fig. 6. The distance between neighboring nodes is 275m, and the transmission range is 280m. The other simulation parameters are listed in Table II.

Fig. 7 shows the average competing nodes of four mechanisms with different *XmtHoldoffExponent*. Aware mechanism performs the best, and unaware mechanism performs the worst. Because the enhanced EBTT extends regular *NextXmtTimeInterval* twice, the enhanced EBTT performs a little worse than EEBTT while *XmtHoldoffExponent* is greater than three, but it still performs almost the same as EEBTT if *XmtHoldoffExponent* remains small.

The interval between each DSCH message is the key factor to determine the access time of the control message. Fig. 8 is the results of the average DSCH messages interval with different *XmtHoldoffExponent*. In this figure, all four mechanisms perform almost the same. Since the DSCH messages interval is combined of *XmtHoldoffTime* and the number of the slots lost in the competition, the fact that *XmtHoldoffTime* increases exponentially by *XmtHoldoffExponet* makes the value of *XmtHoldoffTime* dominate the average DSCH messages interval.

Fig. 9 is the error ratio of the DSCH messages caused by unexpected collisions. All mechanisms except EEBTT experience zero error ratio during the simulation. Although the error ratio of EEBTT is less than 0.5%, it still could have a significant impact on the time-sensitive traffic.

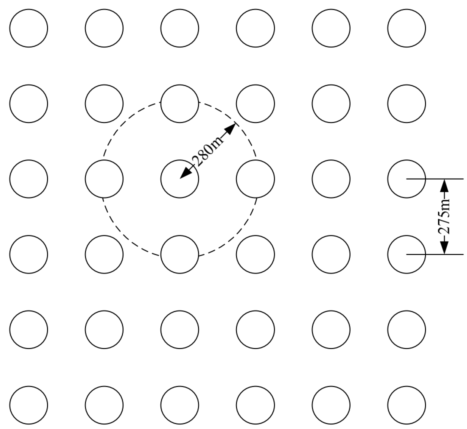


Figure 6. The simulated topology.

TABLE II
SIMULATION PARAMETERS

| Parameter | Value |
|---|---|
| Frame length | 4ms |
| MSH_CTRL_LEN | 4 |
| MSH_DSCH_NUM | 4 |
| Scheduling Frames | 4 |
| Bandwidth | 10MHz |
| Simulation duration | 100s |

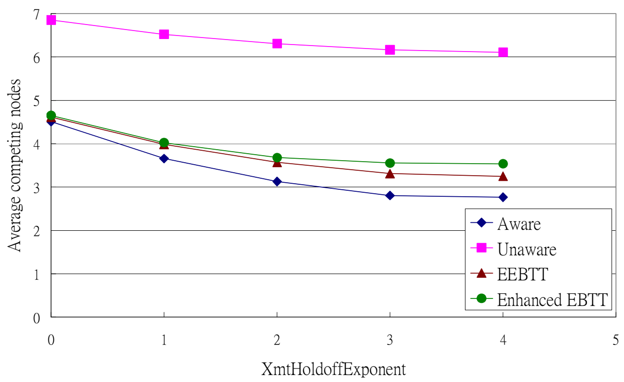


Figure 7. Average competing nodes VS XmtHoldoffExponent.

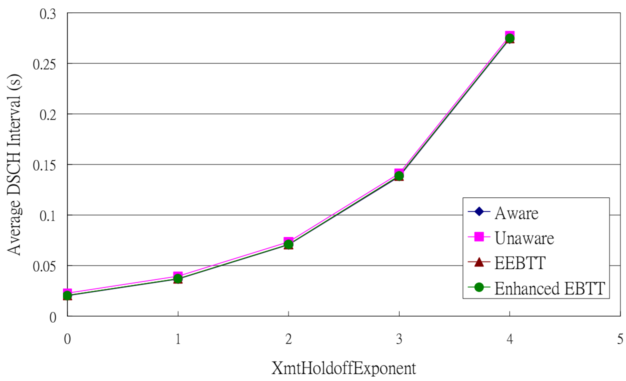


Figure 8. Average DSCH Interval VS XmtHoldoffExponent.

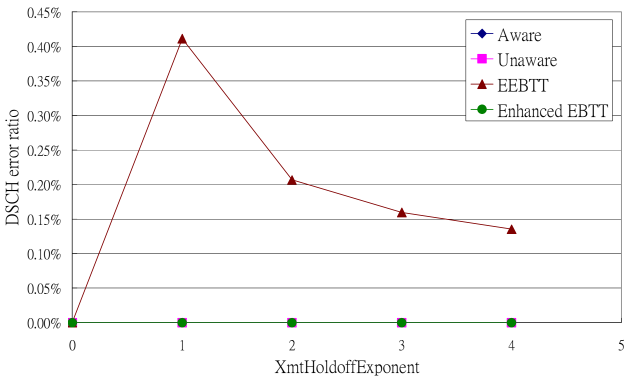


Figure 9. DSCH error ratio VS XmtHoldoffExponent.

## V. CONCLUSIONS

We describe the coordinated distributed scheduling algorithm of IEEE 802.16 and the control message transmission timing mechanism in this paper. We also state the problem that causes unexpected collision on the current mechanism. The enhanced EBTT mechanism is proposed to eliminate these collisions. The simulation results show that even the enhanced EBTT mechanism performs a little worse compared to EEBTT, it completely avoids colliding on the MSH-DSCH messages.

The future work will consider QoS by giving the higher priority nodes a better chance to transmit their MSH-DSCH. There are a few ways to achieve it. For example, adapting the *XmtHoldoffExponent* to shorten the average MSH-DSCH interval or change the *XmtHoldoffTime* regardless of the IEEE 802.16 standard. The topic about how to combine the centralized and distributed schedulings is also useful to guarantee QoS since it is not efficient to make a multi-hop connection via distributed scheduling, but it is also not suitable to do a neighboring node data switching via the centralized scheduling. The adaption of switching between the centralized and distributed schedulings is an interesting issue and remains unknown recently.

## REFERENCES


[1] Raffaele Bruno, Marco Conti, and Enrico Gregori, “Mesh Networks: Commodity Multihop Ad Hoc Networks,” IEEE Communications Magazine, Mar. 2005.

[2] Roberto Hincapie, Javier Sierra, and Roberto Bustamante, “Remote Locations Coverage Analysis with Wireless Mesh Networks based on IEEE 802.16 Standard,” IEEE Communications Magazine, Jan. 2007.

[3] IEEE 802.16-2004, “IEEE standard for Local and Metropolitan Area Networks — Part 16: Air Interface for Fixed Broadband Wireless Access Systems,” Oct. 1, 2004.

[4] Nico Bayer, Dmitry Sivchenko, Bangnan Xu, Veselin Rakocevic, and Joachim Habermann, “Transmission timing of signaling messages in IEEE 802.16 based Mesh Networks,” European Wireless 2006.

[5] Dave Beyer, Nico van Waes, and Carl Eklund, “Tutorial: 802.16 MAC Layer Mesh Extension Overview,” March 2002. https://www.wirelessman.org

[6] “https://www.isi.edu/nsnam/ns/”

[7] Claudio Cicconetti, Ian F. Akyildiz, and Luciano Lenzini, “Bandwidth Balancing in Multi-Channel IEEE 802.16 Wireless Mesh networks,” IEEE INFOCOM 2007.