An Adaptive Holdoff Algorithm based on Node State for IEEE 802.16 Mesh Mode with Coordinated Distributed Scheduling

Bong Chan Kim, Dong Gu Kwak, Heecheol Song, Hwang Soo Lee, and Joong Soo Ma*
School of Electrical Engineering and Computer Science, KAIST, Daejeon, South Korea
*School of Engineering, ICU, Daejeon, South Korea
{eeman, dgkwak, practice00}@mcl.kaist.ac.kr, hwanglee@ee.kaist.ac.kr, jsma@icu.ac.kr

Abstract—Wireless mesh networks (WMNs) are one of the key features of beyond 3G system because of their flexible and low-cost deployment. IEEE 802.16 mesh mode has recently emerged as an alternative protocol for establishing WMN. Because the WMN is used as wireless backbone network, the design of WMN protocol should target a high network throughput. Then, in IEEE 802.16 mesh standard, the performance improvement is limited by the holdoff algorithm in the bandwidth reservation procedure. In this paper, we propose an adaptive holdoff algorithm for IEEE 802.16 mesh mode with coordinated distributed scheduling. The proposed holdoff algorithm improves network throughput by adaptively adjusting the transmission holdoff exponent according to current node state. In addition, the proposed holdoff algorithm maintains the backward compatibility with the IEEE 802.16 mesh standard and does not require any information from other layers. Simulation results show that the proposed holdoff algorithm performs better than existing holdoff algorithm in terms of total network throughput.

Keywords—IEEE 802.16 mesh mode, holdoff algorithm, distributed scheduling, wireless mesh network

I. INTRODUCTION

Wireless mesh networks (WMNs) are one of the key features of beyond 3G system because of their flexible and low-cost deployment. IEEE 802.16 mesh mode [1] has recently emerged as an alternative protocol for establishing wireless mesh networks.

In the IEEE 802.16 mesh mode with a coordinated distributed scheduling (C-DSCH) [2], multi-hop communication is possible between mesh base stations (MeshBSs) and mesh subscriber stations (MeshSSs) because all nodes are peers and each node can act as routers to support multi-hop packet forwarding [3].

In the IEEE 802.16 mesh with C-DSCH, every node competes for channel access using a distributed election algorithm (DEA) based on the scheduling information of the extended neighborhoods (one-hop and two-hop neighbors). IEEE 802.16 mesh with C-DSCH defines a three-way handshaking mechanism that uses a mesh distributed scheduling (MESH-DSCH) message to request, grant, and confirm available data-subframe. If the interval between two subsequent MSH-DSCH messages of a node is long, the three-way handshaking takes very long, and packets get a long queuing delay [3].

Some holdoff algorithms for the IEEE 802.16 mesh with C-DSCH have been proposed to improve the capacity of mesh network by reducing interval between two subsequent MSH-DSCH messages. The static holdoff algorithm is described in the IEEE 802.16 mesh standard [1]. In the static holdoff algorithm, because all MeshSSs and MeshBSs have an identical transmission holdoff exponent (Xmt_Holdoff_Exp), they have an identical transmission holdoff opportunity (Xmt_Holdoff_Opp) regardless of current node state. Therefore, when Xmt_Holdoff_Exp = 0, the competition between nodes happens severe, and a node experiences several times failures before reserving bandwidth. As the Xmt_Holdoff_Exp increases, nodes get a longer holdoff time, and thus the contention between nodes becomes less competitive. However, the longer Xmt_Holdoff_Opp results in a long queuing delay [4].

In [4], the authors performed the modeling and performance analysis of distributed scheduler in IEEE 802.16 mesh mode, and they concluded that the capacity of IEEE 802.16 mesh network can be optimized by assigning appropriate exponent values to nodes in the network.

In [5], the dynamic exponent (DynExp) algorithm was proposed. In the DynExp algorithm, nodes that are currently not sending, receiving, or forwarding data packets use large Xmt_Holdoff_Exp. Nodes that transmit, receive, and forward data packets or nodes that have been selected by the routing protocol as potential forwarding nodes use small Xmt_Holdoff_Exp. By setting different Xmt_Holdoff_Exp values to node classifications, the DynExp improves network capacity. Then, because the operation of DynExp algorithm depends on information from the routing layer, the DynExp cannot be operated independently.

In this paper, we propose an adaptive holdoff algorithm for IEEE 802.16 mesh mode with C-DSCH. The proposed holdoff algorithm improves the network throughput by adaptively adjusting the Xmt_Holdoff_Exp according to current node state. In addition, it maintains the backward compatibility with the IEEE 802.16 mesh standard and does not require any information from other layers.
The remaining sections of the paper are presented as follows; In Section II, the IEEE 802.16 mesh mode with C-DSCH is described briefly. The neighborhoods management is described in detail in Section III. The proposed adaptive holdoff algorithm is outlined in Section IV. In Section V, we describe the simulation environments used to evaluate the performance of the proposed holdoff algorithm and provide the simulation results in Section VI. Finally, conclusions are presented in Section VII.

II. IEEE 802.16 MESH MODE WITH C-DSCH [2][3][5]

Figure 1 shows the frame structure of IEEE 802.16 mesh mode. As shown in the figure, the frame of IEEE 802.16 mesh mode is based on the time division multiple access (TDMA) frame structure and it consists of the control-subframe and the data-subframe. The data-subframe is used to transmit data packet, and the control-subframe is used to transmit MAC management messages. The control-subframe and data-subframe are fixed in length and consist of multiple transmission opportunities (TOs) and minislots, respectively.

The control-subframe is classified into two types: the network-control-subframe and the schedule-control-subframe, which are not shown in Fig. 1. The network-control-subframe is used to broadcast network entry/configuration information to all MeshSSs, and it enables a new MeshSS to gain synchronization and initial network entry into a mesh network. The schedule-control-subframe is used to transmit MAC management messages for reserving the minislots of the data-subframe, and it is divided into two parts; the first part is for the centralized scheduling mechanism (CSCH), and it is used to send the mesh CSCH (MSH-CSCH) and mesh centralized configuration (MSH-CCFG) messages. The second part is for the C-DSCH, and the MSH-DSCH messages are transmitted over this part.

In this section, we describe only the IEEE 802.16 mesh mode with C-DSCH from the three-way handshaking perspective. In the C-DSCH, every node sends its available resource information to neighbor nodes via MSH-DSCH messages. As shown in Fig. 2, A MSH-DSCH message contains the following information elements (IEs): The Scheduling_IE includes the next MSH-DSCH transmission times and holdoff exponents of a node and its neighbor nodes. The Request_IE is used by a node to specify its bandwidth demand for a specific link. The Availability_IE is used by a node to convey the state of individual minislots to its neighbors. The Grant_IE is used by a node to send grants in response to a bandwidth request as well as to send a confirmation (grant confirmation) for a received bandwidth grant.

Based on the above IEs, in the C-DSCH, the bandwidth reservation is accomplished by the three-way handshaking: bandwidth request, bandwidth grant, and bandwidth grant confirmation, as shown in Fig. 3. The more detailed explanation of the three-way handshaking refers to IEEE 802.16 mesh standard [1].

In the three-way handshaking procedure, the transmission time of MSH-DSCH message is selected by the DEA. The DEA enables nodes to forward messages in a multi-hop mesh network without explicit schedule negotiation. The transmission time obtained by the DEA is collision-free within the two-hop neighborhoods of each node. Furthermore, the DEA is operated in a completely distributed manner and needs no central controller. In the C-DSCH, the transmission time corresponds to a specific TO.

Every node should inform its neighbors about next MSH-DSCH transmission times of it as well as its one-hop neighbors. Every node can calculate the next transmission times of all nodes within its two-hop distance, and thus it can avoid collision in MSH-DSCH message transmission. After the last transmission time, a node is not eligible to transmit a MSH-DSCH message for transmission holdoff time (H) in order to share radio resource between mesh nodes.

\[
H = 2^{Xmt\_Holdoff\_Exp\_Base + Xmt\_Holdoff\_Exp}\quad (1)
\]

In the IEEE 802.16 standard, the Xmt_Holdoff_Exp_Base is fixed to 4, and the Xmt_Holdoff_Exp is selected to a constant in the range between 0 and 7 in the node’s
The neighbor table is managed as follows:

1) Setting Type_Tx/Rx, Expire_Tx/Rx, and Expire:

Whenever a node sends or receives a data packet to or from a neighbor, it finds the entry for that neighbor in its neighbor table and sets Type_Tx, Expire_Tx, and Expire as follows: Type_Tx is set to 1. Expire_Tx and Expire are set to EXPIRE_TIME. When a node sends a data packet to a neighbor, it finds the entry for that neighbor in its neighbor table and sets Type_Rx, Expire_Rx, and Expire as follows: Type_Rx is set to 1. Expire_Rx and Expire are set to EXPIRE_TIME.

2) Maintaining/removing neighbor table entries:

Maintenance and removal of a neighbor table entry is performed using the neighbor timer. The neighbor timer expires periodically; when this occurs, a node compares the current time with each value of Expire_Tx/Rx and Expire in a neighbor table entry and updates Type_Tx/Rx, Expire_Tx/Rx, and Expire. If the values of Expire_Tx/Rx are higher than the current time, the values of Type_Tx/Rx and Expire_Tx/Rx are not changed; otherwise, they are set to 0. If the value of Expire is less than the current time, the neighbor table entry is removed; otherwise, the value of Expire is not changed. This procedure is repeated for all neighbor table entries.

Figure 5 shows the neighbor table of node I in Fig. 4. By checking the values of Type_Tx/Rx in neighbor table entries, it can be verified that node currently has at least one tx-neighbor or at least one rx-neighbor.

### III. NEIGHBORHOODS MANAGEMENT

Before proposing the adaptive holdoff algorithm, we introduce a neighborhoods management. First, the neighbors of a node are classified into four types: tx-neighbor, rx-neighbor, tx/rx-neighbor, and null-neighbor. Each classification is defined as follows: A tx-neighbor of a neighbor that has a data packet to send to the node. An rx-neighbor of a neighbor that has no data packet to send to or receive from the node. In Fig. 4, node I has four types of neighbors. Nodes A, B, C, and D correspond to the tx-neighbor, rx-neighbor, tx/rx-neighbor, and null-neighbor of node I, respectively.

To classify neighbors into four types, nodes store neighbor information in their neighbor table. Figure 5 shows the neighbor table, which is used in the proposed holdoff algorithm later. Each entry in the neighbor table contains Node_ID, Type_Tx/Rx, Expire_Tx/Rx, and Expire. Node_ID indicates the identifier of a neighbor. Type_Tx/Rx is used to distinguish the neighbor types: tx-neighbor, rx-neighbor, tx/rx-neighbor, or null-neighbor. Expire_Tx/Rx indicates the expiration time of a neighbor as a tx-neighbor or an rx-neighbor. Expire represents the expiration time of a neighbor table entry.

Neighbor table is managed as follows:

1) Setting Type_Tx/Rx, Expire_Tx/Rx, and Expire:

When a node sends or receives a data packet to or from a neighbor over minislots of data-subframe, it sets the Type_Tx/Rx, Expire_Tx/Rx, and Expire in the neighbor table entry for the neighbor. When a node receives a data packet from a neighbor, it finds the entry for that neighbor in its neighbor table and sets Type_Tx, Expire_Tx, and Expire as follows: Type_Tx is set to 1. Expire_Tx and Expire are set to EXPIRE_TIME. When a node sends a data packet to a neighbor, it finds the entry for that neighbor in its neighbor table and sets Type_Tx, Expire_Tx, and Expire as follows: Type_Tx is set to 1. Expire_Tx and Expire are set to EXPIRE_TIME. When a node sends a data packet to a neighbor, it finds the entry for that neighbor in its neighbor table and sets Type_Tx, Expire_Tx, and Expire as follows: Type_Tx is set to 1. Expire_Tx and Expire are set to EXPIRE_TIME. When a node sends a data packet to a neighbor, it finds the entry for that neighbor in its neighbor table and sets Type_Tx, Expire_Tx, and Expire as follows: Type_Tx is set to 1. Expire_Tx and Expire are set to EXPIRE_TIME.

![Fig. 4 Classification of neighbor nodes: A solid line between a pair of nodes indicates that there is at least one data packet to transmit between the two nodes, with an arrow showing the direction of the pending data transmission. A dotted line indicates that there is no data waiting for transmission between the pair of nodes.](image1)

<table>
<thead>
<tr>
<th>Node_ID</th>
<th>Type_Tx</th>
<th>Type_Rx</th>
<th>Expire_Tx</th>
<th>Expire_Rx</th>
<th>Expire</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>0</td>
<td>nonzero</td>
<td>0</td>
<td>nonzero</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>nonzero</td>
<td>nonzero</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>1</td>
<td>nonzero</td>
<td>nonzero</td>
<td>nonzero</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>nonzero</td>
</tr>
</tbody>
</table>

![Fig. 5 Neighbor table.](image2)

In this section, we propose an adaptive holdoff algorithm based on node state. Based on the classification of neighbors mentioned to Section III, we first define the Weight of a node as follows:

\[ Weight = tx\_exist \times 1 + rx\_exist \times 2 \] (2)

The Weight of a node represents its own competing weight in the shared medium. In equation (2), the tx\_exist represents whether a node has at least one tx-neighbor or not. If a node has at least one tx-neighbor, tx\_exist is set to 1; otherwise, tx\_exist is set to 0. The rx\_exist indicates whether a node has at least one rx-neighbor or not. If a node has at least one rx-neighbor, rx\_exist is set to 1; otherwise, rx\_exist is set to 0. As shown in Fig. 2, the MSH-DSCH message can contains multiple MSH-DSCH Request_IE(s), MSH-DSCH Availability_IE(s), and MSH-DSCH Grants_IE(s) at one time. That is, node can simultaneously request, grant, and confirm bandwidth for multiple links via only one MSH-DSCH message. Thus, the Weight depends on only tx\_exist and rx\_exist.
Algorithm 1 Adaptive Holdoff Algorithm

Notations Used:
\( tx\_exist \) = indicate whether a node currently has at least one tx-neighbor
\( rx\_exist \) = indicate whether a node currently has at least one rx-neighbor
\( Weight \) = competing weight of a node
\( Current\_Xmt\_Holdoff\_Exp \) = current transmission holdoff exponent
\( Xmt\_Holdoff\_Exp\_Base \) = transmission holdoff exponent base
\( Max\_Xmt\_Holdoff\_Exp \) = maximum value of transmission holdoff exponent (=3)
\( H \) = transmission holdoff time

01: Set \( tx\_exist \) by checking the Type_Txs of neighbor table entries
02: Set \( rx\_exist \) by checking the Type_Rxs of neighbor table entries
03: Compute \( Weight = tx\_exist \times 1 + rx\_exist \times 2 \)
04: if (\( Weight = 0 \))
05: \( Current\_Xmt\_Holdoff\_Exp = Max\_Xmt\_Holdoff\_Exp \)
06: end if
07: else if (\( 0 < Weight \leq 3 \))
08: \( Current\_Xmt\_Holdoff\_Exp = Max\_Xmt\_Holdoff\_Exp - Weight \)
09: end else if
10: else
11: \( Current\_Xmt\_Holdoff\_Exp = Max\_Xmt\_Holdoff\_Exp \)
12: end else
13: Compute \( H = 2^{Xmt\_Holdoff\_Exp\_Base - Current\_Xmt\_Holdoff\_Exp} \)

TABLE 1 Simulation Parameters used in Holdoff Algorithms

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. transmission range</td>
<td>MeshSS: 400m, MeshBS: 400m</td>
</tr>
<tr>
<td>Bytes per OFDM symbol</td>
<td>10B (64-QAM 3/4)</td>
</tr>
<tr>
<td>Xmt_Holdoff_Exp_Base</td>
<td>4</td>
</tr>
<tr>
<td>Xmt_Holdoff_Exp</td>
<td>3 (only static holdoff algorithm)</td>
</tr>
<tr>
<td>Distance between MeshSSs</td>
<td>300m</td>
</tr>
</tbody>
</table>

When a node reserves bandwidth for data transmission to its \( rx\)-neighbor, it is required that the node sends two MSH-DSCH messages (bandwidth request and confirm) during the three-way handshaking. Then, when a node reserves bandwidth for data reception from its \( tx\)-neighbor, the node sends only one MSH-DSCH message (bandwidth grant) to its \( tx\)-neighbor in the bandwidth reservation procedure. In this context, the \( tx\_exist \) and \( rx\_exist \) are multiplied by 1 and 2 in equation (2), respectively. For example, the Weight of node 1 in Fig. 4 is equal to 3 (= \( 1 \times 1 + 1 \times 2 \)).

In an adaptive holdoff algorithm, the Xmt_Holdoff_Exp of a node is adjusted according to its Weight. Algorithm 1 shows the operation of the adaptive holdoff algorithm. In Algorithm 1, suppose that the Xmt_Holdoff_Exp_Base is set to 4 and the Max_Xmt_Holdoff_Exp to 3.

As shown in Algorithm 1, if a node has a large Weight, the small \( H \) is obtained from a small Cur_Xmt_Holdoff_Exp. Thus, the node can access the TO of schedule-control-subframe for bandwidth reservation at a high rate. On the other hand, if a node has a small Weight, the large \( H \) is obtained from a large Cur_Xmt_Holdoff_Exp. Thus, the node can access the TO of schedule-control-subframe for bandwidth reservation at a low rate. In this way, nodes can adjust their access rate to the TO of schedule-control-subframe according to their current state (Weight), and thus the network performance improves.

V. SIMULATION ENVIRONMENTS

Computer simulations are performed to evaluate the performance of the adaptive holdoff algorithm (AHA) via NCTUs-4.0 [6][7]. The performance of the proposed holdoff algorithm is compared with that of the static holdoff algorithm (SHA), which is described in the IEEE 802.16 mesh standard.

TABLE 1 shows the simulation parameters used in the simulation. For all other parameters, the default values provided in NCTUs-4.0 are used. Figure 6 shows the network topology used in the simulation. The network consists of one MeshBS and 15 MeshSSs. Node 11 corresponds to the MeshBS and other nodes to MeshSSs. All nodes (MeshSS and MeshBS) remain stationary for a simulation time of 300s. The number of traffic flows is varied from 1 to 6 to investigate the performance variation in different offered loads. Each traffic source generates UDP data packets with the size of 512 bytes at a rate of 2Mbits/s.

Simulations are performed in two scenarios: multi-hop peer-to-peer and multi-hop Internet access scenarios. The traffic flows for the multi-hop peer-to-peer scenario are shown in Fig. 7. The multi-hop Internet access scenarios are classified into upload and download patterns. The traffic flows for upload and download patterns are shown in Figs. 8 and 9, respectively.
To evaluate the performances of the holdoff algorithms, total network throughput is used as performance metric. The total network throughput is the sum of the average packet throughputs of all receivers.

VI. SIMULATION RESULTS

In the multi-hop peer-to-peer scenario where traffic is distributed throughout entire network, the performance of AHA is compared with that of SHA. Figure 10 shows the total network throughput. In the static holdoff algorithm, because all node have an identical Xmt_Holdoff_Exp (= 3), they have an identical holdoff time regardless of their current state. However, in the adaptive holdoff algorithm, because a node adjusts its Xmt_Holdoff_Exp in a range between 0 and 3 according to the current state, the bandwidth can be used efficiently.

For example, because a node with Weight = 0 currently has no data communication with neighbors, even though the node has the Xmt_Holdoff_Exp of 3, the communication problem does not happen. Furthermore, because the node reduces the access rate to schedule-control-subframe, other nodes can access schedule-control-subframe at a high rate. On the other hand, because a node with Weight = 3 currently has data communication with neighbors, it can access schedule-control-subframe at a high rate by setting Xmt_Holdoff_Exp to 0. By adaptively adjusting the Xmt_Holdoff_Exp according to the current node state, the AHA performs better than the SHA in terms of total network throughput, as shown in Fig. 10.

In general, access to the Internet through MeshBS is desirable for MeshSSs to obtain necessary service for survival or business. In the multi-hop Internet access scenario, it frequently happens that MeshSSs upload files to Internet through MeshBS and MeshSS download files provided in the Internet through MeshBS. As shown in Figs. 8 and 9, computer simulations are performed in the multi-hop Internet access scenario with upload and download patterns. Figure 11 shows total network throughput in the multi-hop Internet access scenario with upload pattern. In addition, Fig. 12 shows total network throughput in the multi-hop Internet access scenario with download pattern. In the multi-hop Internet access scenario, it frequently happens that nodes act as one of sender or receiver. For example, node 12 serves only as sender in Fig. 8, and only as receiver in Fig. 9. In the AHA, because a node sets its Xmt_Holdoff_Exp to an appropriate value according to the current role, the AHA outperforms the SHA, as shown in Figs. 11 and 12.

VII. CONCLUSION

In this paper, we proposed an adaptive holdoff algorithm for IEEE 802.16 mesh mode with C-DSCH. The proposed holdoff algorithm improves network throughput by adaptively adjusting the Xmt_Holdoff_Exp according to current node state (Weight). In addition, the adaptive holdoff algorithm maintains the backward compatibility with the IEEE 802.16 mesh standard and does not require any information from other layers. Simulation results showed that the proposed holdoff algorithm performs better than existing holdoff algorithm in terms of total network throughput.

ACKNOWLEDGMENT

This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) and MMPC (Mobile Media Platform Center) support programs supervised by the IITA (Institute of Information Technology Advancement)

REFERENCES