An Optimal Scheduling Algorithm for Maximizing Throughput in WiMAX Mesh Networks

Chih-Yung Chang, Ming-Hsien Li, Wen-Chuan Huang, Shih-Chieh Lee
Dept. Computer Science and Information Engineering, Tamkang University, Tamsui, Taiwan
*HTC Corporation, Taiwan
cychang@mail.tku.edu.tw, minghсли@ gmail.com, joy@ wireless.cs.tku.edu.tw, aaron@ wireless.cs.tku.edu.tw

Abstract— WiMAX Mesh Network architecture is defined in the IEEE 802.16 standard for increasing network coverage and improving communication performance. In the past few years, several greedy or heuristic algorithms have been proposed to cope with the scheduling problem in WiMAX mesh networks. However, their performance highly depends on the network topology and bandwidth requests and they do not achieve optimal performance in all cases. This paper proposes an optimal scheduling algorithm, called SADP, which exploits the opportunities of spatial reuse and maximizes the network throughput based on the network topology and the uplink bandwidth requests of each Subscriber Station (SS). In addition, a heuristic scheduling algorithm (HSA) is proposed to reduce the computing complexity. The performance results were approximate to the optimal results. Simulation study reveals that the proposed SADP provides the WiMAX mesh network with maximal throughput and shortest transmission time, and the proposed HSA likely achieves the optimal results.

Keywords: WiMAX, 802.16 Mesh Networks, Scheduling, Dynamic Programming, Spatial Reuse.

I. INTRODUCTION

Wireless Metropolitan Area Network (WMAN) provides wireless broadband services for an extensive coverage area. An IEEE 802.16 network consists of one Base Station (BS) and several Subscriber Stations (SSs). The BS functions as a gateway that ensures that each of its serving SSs can access external networks, such as the Internet. Each SS works as an access point that wirelessly connects to a dedicated BS and manages data delivery between the BS and mobile or static terminals.

The IEEE 802.16d standard [1] proposes two modes of frameworks: Point-to-Multipoint (PMP) and Mesh frameworks. In the PMP framework, each SS must directly communicate with the BS; hence, each SS cannot be located more than the Line-of-sight (LOS) distance away from the BS. However, in a mesh framework, a SS is allowed to transmit data to the BS through other SSs in a multi-hop manner. The mesh network architecture extends the coverage of the BS and enables the routing path to be dynamically updated or repaired under unpredicted situations, such as path breakdown or inferior radio qualities for some relay SSs. Consequently, the mesh framework supports higher reliability and superior availability compared to the PMP framework.

The IEEE 802.16 mesh framework supports two resource allocation strategies: distributed scheduling and centralized scheduling. In centralized scheduling, BS is in charge of allocation of bandwidth and arranges the transmission time slots for each SS according to its bandwidth requirement. A tree topology rooted by the BS must be established to describe the routing information between the BS and SSs. In centralized scheduling, the BS periodically collects the network configuration and bandwidth requirement of each SS and arranges a schedule to avoid collision, reduce the transmission cycle, and maximize network throughput. In distributed scheduling, each SS determines its own transmission schedule according to local information, including bandwidth requirements and slot allocation within two-hop neighboring SSs. A contention-based algorithm is required to determine the slot allocation. Prior studies [4]-[6] implemented the distributed resource allocation mechanisms to negotiate flow transmission of neighboring SSs in WiMAX mesh networks. Compared with distributed scheduling, the centralized scheduling mechanism maintains the global information, including the network topology and the bandwidth requirements from all SSs; hence it is more efficient and easier to implement.

According to the IEEE 802.16d standard, two types of control messages are used in the centralized scheduling mechanism: Mesh Centralized Scheduling Configuration (MSH-CSFC) and Mesh Centralized Scheduling (MSH-CSCH). The BS determines a tree topology of the WiMAX mesh network and notifies all SSs by flooding the entire network with the MSH-CSFC packet, which contains the tree topology information. Each SS establishes a link to the parent node in the tree according to the received MSH-CSFC message. The MSH-CSCH is the other control packet, which is composed of two types of messages: Request and Grant. Each SS notifies the BS of its bandwidth requirement by sending the MSH-CSCH:Request message to its parent node. After receiving the bandwidth requirement, the parent node prepares another MSH-CSCH: Request message that contains all bandwidth requirements of its child nodes and its own requirement and sends the message to its parent node. Finally, all MSH-CSCH:Request messages are delivered to the BS along the path defined in the tree topology. After receiving all MSH-CSCH:Request messages, the BS allocates available time slots to each requested SS for transmitting data. Subsequently, the BS notifies all the SSs the decisions, including the bandwidth allocation and scheduling, using a MSH-CSCH:Grant message. The bandwidth allocation
considers the bandwidth requirements from each SS and the child nodes. If the bandwidth requests of SSs change, the SSs inform the BS of their changes of bandwidth requests through the MSH-CSCH messages during the control subframe. After receiving the MSH-CSCH messages, the BS reschedules all transmissions by applying the scheduling algorithm and subsequently rebroadcasts the schedules to all nodes.

Generally, path planning and scheduling are the crucial factors that determine the throughput of a WMN. However, they are not detailed in IEEE 802.16 standards [1]-[3]. In literature, a number of studies [7][8][9] have proposed tree construction algorithms in a WiMAX mesh network. Wei et al. [7] proposed an interference-aware routing protocol that constructs a path from the source node to destination node hop by hop. A source or forwarding node, say a, would select the next forwarding node, say b, from its neighboring nodes if the number of nodes encountering interference from the link (a, b) is minimal. To cope with the network congestion problem, [8] proposed a tree construction protocol for building a routing tree that considers load balance. In [9], a cross layer tree construction protocol that considers both load balance and QoS parameters is proposed. Although previous studies [7][8][9] focused on constructing an efficient tree topology for increasing the network throughput, they did not consider the scheduling problem, which is another crucial factor that determines the network throughput of a WiMAX mesh network.

An efficient scheduling algorithm must consider four factors: collision, congestion, delay, and spatial reuse. Previous studies [10]-[14] considered metrics of spatial reuse and collision to schedule all transmissions with less delay time. Ramanathan and Loyd [15] considered the constraint of deadline delay for supporting QoS transmissions. However, these works [10]-[15] used heuristic or greedy strategies; hence, the scheduling algorithms cannot always provide optimal scheduling. Several studies [16]-[19] have conducted QoS scheduling by considering the required bandwidth and delay over 802.16e PMP networks. However, these scheduling mechanisms are proposed to improve the throughput of the one hop neighbors of the BS and are unsuitable for multi-hop WiMAX mesh networks.

This paper proposes an optimal scheduling algorithm and a heuristic scheduling algorithm that allocates and schedules the available bandwidth for each SS in a mesh network for maximizing the network throughput and minimizing the time required for all transmission requests. Based on the existing mesh network topology, the proposed algorithms allocate each time slot to maximize the network throughput for exploiting the opportunities of spatial reuse. The remainder of this paper is organized as follows: Section II presents the existing scheduling algorithms for WiMAX mesh networks; Section III introduces the proposed optimal scheduling algorithm; Section IV introduces the heuristic scheduling algorithm; Section V provides the performance evaluation of the proposed scheduling algorithm; and finally, Section VI offers a conclusion.

II. RELATED WORK

This section presents a review of the related scheduling mechanisms proposed in [10]-[14] and provides an example (Fig. 1) for comparison with the proposed strategy. Let transmission rate denote the number of bits that are conveyed per second (bits/sec). For simplicity, the example provided in this section assumes that all links in the tree topology have the same transmission rate. A general scheduling algorithm that considers variant rates is proposed in the next section. As shown in Fig. 1(b), the uplink transmission requests required from SS1, SS2, and SS3 are 8, 4, and 3 time slots, respectively. These data are routed along the paths defined in the tree topology and finally arrive at the BS.

![Scheduling tree edge](image1)

(a) An example of the scheduling tree topology.

![Required Data slot](image2)

(b) SS1, SS2, and SS3 require 8, 4 and 3 time slots, respectively.

Fig. 1. A typical example of scheduling problem in IEEE 802.16 Mesh Network.

Based on the tree topology shown in Fig. 1(a), Fig. 2 shows an optimal scheduling that considers both spatial reuse and collision free factories. Let notation (v_i, v_j) indicate the link from v_i to v_j. First, the BS schedules a parallel transmission for v_2 and v_3 and assigns 4 and 3 transmission slots for them, respectively. After receiving data from v_3, v_4 starts to forward data to v_5 at Slot 4. The BS switches to receive data from another neighbor, v_1. During the period of transmission from v_1 to the BS, another transmission (v_3, v_5) progresses simultaneously to increase the spatial reuse. Eventually, the transmission requirements for each SS are satisfied within 15 time slots.

![Optimal Scheduling](image3)

Fig. 2. The optimal scheduling of the example shown in Fig. 1(a).

In [10], Shetyia and Sharma proposed a scheduling algorithm that applies a dynamic programming scheme to achieve maximal throughput within n time slots. For a tree with various transmission rates on the tree links, the proposed scheduling algorithm starts allocating time slots from leaf nodes and ends at the BS. However, this algorithm does not consider spatial reuse; hence, only one SS is assigned to transmit data at each time slot. Because spatial reuse was not considered, various transmission arrangements have the same result in terms of required time slots. We used the sequence (v_1, v_2), (v_2, v_3), (v_3, v_5), (v_2, v_6), (v_1, v_0) to evaluate the total number of time slots. Fig. 3 shows the schedule by applying the algorithm proposed in [10]. The number of time slots required to satisfy all
transmission requirements is 24, which is larger than that required by the optimal scheduling algorithm.

![Fig. 3. The transmission schedule by applying the scheduling algorithm proposed in [10].](image)

Cheng et al. [11] proposed a Combined Distributed and Centralized (CDC) scheme that combines the distributed scheduling and centralized scheduling mechanisms to increase the time slot utilization. Fig. 4 shows the scheduling by applying the scheduling algorithm proposed in [11]. Because the CDC scheme allocates the time slots in an order based on the index of each SS, the link \((v_1, v_0)\) is first allocated 8 time slots. Because \((v_3, v_4), (v_4, v_3)\) and \((v_5, v_2)\) do not interfere with \((v_1, v_0)\), the BS allocates 3, 3, and 2 time slots to links \((v_5, v_4), (v_4, v_3), \) and \((v_3, v_2)\), respectively. Subsequently, the BS allocates 6 time slots to \((v_2, v_0)\). Finally, the BS allocates the bandwidth resource for the data transmission of \(v_3\), and reserves one slot each for \((v_1, v_2)\) and \((v_2, v_0)\). Consequently, the number of time slots is 16, which is not optimal.

![Fig. 4. The scheduling by applying algorithm proposed in [11].](image)

Han et al. [12] and [13] used the concepts of Primary and Secondary interferences to present an interference model for a WiMAX mesh network. Four policies for determining the order of allocating bandwidth resource to nodes in a mesh network are proposed based on various metrics, as follows: Random, Min Interference, Nearest Node to BS First (NNBF), and Farthest Node to BS First (FNBF). The simulation results showed that the policy of NNBF outperforms the other three policies. Therefore, we applied the policy of NNBF to the network shown in Fig. 1 (a) and compared its performance with the result of the optimal scheduling. As demonstrated in [12] and [13], if two SSs have the same hop count to the BS, the BS with the smaller node ID has higher priority for allocation of time slots. The scheduling result by applying [12] and [13] is the same as that shown in Fig. 4. Compared with the optimal scheduling, [12] and [13] require more time slots.

In [14], a heuristic approach was proposed to cope with the admission control and scheduling problems in a WiMAX mesh network. Each link in the network was assigned a label of even or odd according to the layer to which it belongs. An even link transmits in an even time slot, whereas an odd link transmits in an odd timeslot. The algorithm proposed in [14] initially attempts to allocate free time slots and sub-channels to each flow and to allocate maximal bandwidth to all links belonging to the flow. If the allocated bandwidth of a link is more than the minimum requirement of a flow, the assigned time slots to the link can be used by other flows if required. Fig. 5 shows the transmission schedule when applying the algorithm proposed in [14]. The algorithm initially allocates \((v_1, v_0)\) with 8 odd time slots (slots 1, 3, 5, 7, 9, 11, 13, and 15). To avoid interference with flow \((v_4, v_3)\), flow \((v_2, v_0)\) is allocated 2, 4, 6, and 8 slots, which are even slots. Because the links belonging to flows \((v_4, v_3)\) and \((v_1, v_0)\) do not interfere with each other, the BS allocates odd time slots to the links belonging to flow \((v_4, v_3)\). Consequently, transmissions of all flows can be completed at time Slot 23, which is more than the time required for optimal scheduling.

![Fig. 5. The schedule by applying algorithm proposed in [14].](image)

The approaches proposed in [10]-[14] did not completely explore the opportunities of spatial reuse and required more time slots than the optimal scheduling. In the next section, a centralized scheduling algorithm is proposed to arrange an optimal schedule for a WiMAX mesh network. The proposed optimal approach arranges the maximal number of parallel transmissions in each time slot and provides the shortest transmission period for all SSs. Therefore, the proposed optimal approach achieves the shortest delay time and highest throughputs.

### III. OPTIMAL SCHEDULING ALGORITHM

#### A. Network Environment and Problem Definition

This paper proposes two scheduling algorithms in a WiMAX mesh network \(G = (V, E)\), where \(V\) and \(E\) denote the set of nodes and the set of links in the mesh network, respectively. The node set \(V = \{v_i\, 0 \leq i \leq n\}\) comprises \(n+1\) nodes, where \(v_0\) represents the BS and \(\{v_1, v_2, ..., v_n\}\) represents the \(n\) SSs. Let \(T = (V, E', R)\) denote the scheduling tree for a given WMN \(G = (V, E)\), where \(E' \subseteq E\) is the set of links in the scheduling tree and \(R\) is the set of link transmission rates. Let \(l_{ij}\) denote the link between \(v_i\) and \(v_j\) where \(v_i\) is the parent of \(v_j\) in tree \(T\) and \(r_{ij}\) denotes the data transmission rate of \(l_{ij}\). Let \(d_i\) denote the bandwidth request of node \(v_i\), which is the amount of data required for transmission per second (bits/sec). The set of bandwidth requests maintained by the BS is denoted by \(D = \{d_1, d_2, ..., d_n\}\). According to the scheduling tree \(T\) and request set \(D\), the BS makes an optimal schedule that assigns the maximal number of transmissions in each slot to ensure minimal transmission time required for satisfying the requests of \(D\).

To explore the opportunities of parallel transmissions, we used the Primary and Secondary interference relations proposed in [12]. The primary interference indicates that a node cannot transmit and receive data simultaneously. This constraint is also referred to as the transmission/reception constraint. The secondary interference indicates that all neighboring nodes, except for the sender, of a receiver...
cannot transmit data at the time slot when the receiver receives data from the sender. This constraint is also referred to as the interference-free constraint. Based on the concepts of primary and secondary interference constraints, a two-dimensional Collision Matrix (CM) is used to specify the interference relations between nodes in the network. The value of entry CM(i, j) is defined by

\[ \text{CM} (i, j) = \begin{cases} 0 & \text{, otherwise} \\ 1 & \text{, if } v_i \text{ interferes with } v_j. \end{cases} \]

We assumed that the BS knows the CM information because it has collected MSH-NCFG [1] messages, including the neighbor information of each SS. Consequently, for a given set of SSs, the BS can use CM to determine nodes that can be arranged at the same slot. Mathematically, the problem and objectives of this study were formulated as follows:

In a mesh network, a data flow is defined as a sequence of packets from a source node to a destination. Let \( p_{i,a} = \{v_i = v_i^0, v_i^1, \ldots, v_i^{a-1}, v_a\} \) denote the path of flow from \( v_i \) to \( v_a \). For a given tree \( T \) and a request set \( D \), a working set \( W_{T,D} \) denotes the set of all SSs that participate in the data transmission for satisfying all requests in \( D \) that is, \( W_{T,D} = \bigcup_{i \in D} p_i \). A scheduling \( S_{T,D} \) is valid if a set of time slots \( T_i \) is assigned to node \( v_i \) to satisfy all requests \( d_i \in D \) for all \( v_i \in W_{T,D} \). The goal of this study was to find an optimal scheduling \( S_{opt} \) to ensure that scheduling \( S_{opt} \) is valid and the number of total required time slots is minimized. Let \( t_i \) denote the number of slots \( T_i \) in \( S_{T,D} \). The objective of this study is as follows:

Objective Function:

\[
\text{Minimize } t = \max_{v_i \in D} t_i
\]

Subject to the collision free and flow constraints:

- **Collision free constraint:**
  \[ T_i \cap T_j = \emptyset, \text{ if } CM(i, j) = 1 \]

  where \( CM(i, j) \) is the collision relation between nodes \( v_i \) and \( v_j \). The value of entry \( CM(i, j) \) is one or zero, which indicates whether nodes \( v_i \) and \( v_j \) have a collision relation. If the value of \( CM(i, j) \) is 1, nodes \( v_i \) and \( v_j \) have collision relation and cannot be assigned a common slot in their schedules. The collision-free constraint guarantees that the schedules of \( T_i \) and \( T_j \) do not collide.

- **Flow constraint:**
  For any two nodes \( v_i^a \) and \( v_j^b \) in \( P_{i,a} \), let \( t_i^a \) and \( t_j^b \) be the time slots assigned to \( v_i^a \) and \( v_j^b \) for transmitting the same data, respectively.

  \[ t_i^a < t_j^b, \text{ if } j < j' \]

  The condition \( j < j' \) indicates that node \( v_j \) is closer to the source node \( v_i \) than node \( v_j \). Therefore, each packet of a data flow arrives at node \( v_j \) earlier than node \( v_j \). Therefore, the flow constraint requests that the slot assignment must also follow the order of the packet arrivals.

**B. Basic Concepts and Scheduling Rules**

This section presents the basic concept and the scheduling rules that are applied in the proposed scheduling algorithms. In a mesh network, it is difficult for BS to schedule parallel transmissions of multiple flows by considering both collision-free and flow constraints. To guarantee that both constraints can be implemented, the sequential, parallel and nonparallel relations between SSs are identified. According to the IEEE 802.16d standard, the BS is aware of the network topology. The BS subsequently determines the tree topology based on the network topology. According to the network and tree topologies, the BS can obtain the relations between SSs. The sequential relation is defined for nodes belonging to the same flow, whereas parallel and nonparallel relations are defined for nodes belonging to different flows. Let nodes \( v_i \) and \( v_j \) belong to the same flow. The sequential relation is defined as follows:

**Definition: Sequential relation \( \rightarrow \)**

Nodes \( v_i \) and \( v_j \) are said to be sequential and denoted by \( v_i \rightarrow v_j \) if \( v_j \) is closer to the destination than \( v_i \). □

In another word, if \( v_i \rightarrow v_j \), node \( v_j \) will help to forward the traffic from \( v_i \). The transmission time allocated to \( v_j \) must be later than that allocated to \( v_i \) to ensure that the data of \( v_i \) is successfully transferred to the BS. For any nodes \( v_i \) and \( v_j \) belonging to a data flow, the schedule of nodes \( v_i \) and \( v_j \) must satisfy the relation \( v_i \rightarrow v_j \), which indicates that the schedule satisfies the flow constraint.

Let nodes \( v_i \) and \( v_j \) belong to different flows. The parallel relation between nodes \( v_i \) and \( v_j \) is defined as follows:

**Definition: Parallel relation \( \parallel \)**

Node \( v_i \) is said to be parallel with \( v_j \) and denoted as \( v_i \parallel v_j \) if they can transmit data at the same time slot without interference at their receiver sides; otherwise, node \( v_i \) has a nonparallel relation with node \( v_j \) and is denoted by symbol \( v_i \parallel v_j \). □

According to the sequential and parallel relations between SSs, the BS can arrange a valid order of SSs for data transmission in the mesh network.

The following definition extends the relations between two nodes to the relations between two sets.

**Definition: Sequential Relation \( S_a \rightarrow S_b \)**

Two disjoint sets \( S_a \) and \( S_b \) have a Sequential Relation and are denoted by \( S_a \rightarrow S_b \) if they satisfy \( v_i \rightarrow v_j \) for \( \forall v_i \in S_a \) and \( \forall v_j \in S_b \).

If two disjoint sets \( S_a \) and \( S_b \) have a sequential relation, the time slots assigned to \( v_i \in S_a \) and \( v_j \in S_b \) cannot be overlapped. This occurs because each \( v_i \in S_a \) must wait for the data transmitted from each \( v_j \in S_b \), hence, the time slots assigned to \( v_j \) must be later than the time slots assigned to each \( v_i \in S_a \). Fig. 6 shows an example of scheduling that considers the flow from \( v_3 \) to the BS in Fig. 1(a). In this example, both \( v_4 \) and \( v_7 \) have a sequential relation with \( v_3 \), because \( v_2 \) is closer to the BS than \( v_4 \) and \( v_5 \) in the flow. Therefore, the sequential relation \( S_a \rightarrow S_b \) occurs, where \( a = \{v_3, v_4\} \) and \( b = \{v_2\} \). A valid scheduling occurs when the
time slots assigned to $v_2$ are later than the slots assigned to $v_3$ and $v_4$, because $v_2$ must wait for the data sent from $v_3$ and $v_4$.

![Fig. 6. A valid scheduling which considers the flow from $v_3$ to BS in Fig. 1(a).](image)

Assuming that $S_a$ and $S_b$ do not have a sequential relation, a common time slot can be assigned to $v_j \in S_a$ and $v_j \in S_b$ if a parallel relation occurs between $v_j$ and $v_k$. The parallel relation between two sets is defined as follows:

**Definition:** Parallel Relation $P_a \parallel P_b$

Two disjoint sets $P_a$ and $P_b$ have a Parallel Relation and are denoted by $P_a \parallel P_b$ if they satisfy $v_j \parallel v_j$ for $\forall v_j \in P_a$ and $\forall v_j \in P_b$.

Fig. 7 shows a parallel schedule for nodes $v_1$, $v_2$, $v_3$, and $v_5$ in Fig. 1(a). There exists a parallel relation $P_a \parallel P_b$, where $P_a = \{v_1\}$ and $P_b = \{v_3, v_4, v_5\}$.

![Fig. 7. A Parallel scheduling for the example given in Fig. 1(a).](image)

In addition, as shown in Fig. 1(a), a nonparallel relation $N_a \not\parallel N_b$ occurs, where $N_a = \{v_1\}$ and $N_b = \{v_2\}$. Therefore, the BS cannot assign a common slot to both $v_1$ and $v_2$ for satisfying the interference constraint between $v_1$ and $v_2$. Two valid schedules that consider the nonparallel relation between $v_1$ and $v_2$ are shown in Figs. 8(a) and 8(b). No overlapping occurs between the slots assigned to $v_1$ and $v_2$.

![Fig. 8. The scheduling cases consider nonparallel relation for the example given in Fig. 1(a).](image)

These relations may change because of a new node entry or node failure. If any of these situations occur, the SSSs notify the BS of the neighbor change information through the MSH-NCFG [1] messages. The BS can subsequently determine the new tree topology and rebroadcast it to all SSSs. After receiving the new bandwidth requests, the BS reschedules the transmission of all nodes by considering the new relations, which are calculated based on the new network and tree topologies. Based on the sequential, parallel, and nonparallel relations, the proposed algorithm creates an optimal schedule for maximizing the bandwidth utilization and minimizing the total transmission time.

### C. The Scheduling Algorithm with Dynamic Programming Approach (SADP)

This section introduces the proposed Scheduling Algorithm with Dynamic Programming Approach (SADP). The proposed algorithm initially schedules each data flow from the source node to the BS by considering the sequential relation. Subsequently, based on the dynamic programming approach, the SADP algorithm merges two different optimal schedules to derive a larger optimal schedule by arranging the transmission order. A large number of possible transmission sequences may occur during the derivation of the larger optimal schedule. The proposed SADP retains the optimal schedule with minimal transmission time. It is worthy to notice that the merging process must satisfy three relations: sequential, nonparallel, and parallel. The SADP repeatedly executes the same merging process until the optimal schedule that contains all data flow is obtained. Because the SADP must record the optimal schedule of each merging process, a structure, denoted by $U$, represents the schedule.

For a given mesh network, the proposed SADP initially schedules each SS to satisfy its bandwidth requirements and guarantee the shortest transmission period. Let $d_i$ denote the bandwidth request of node $v_i$ and $U(\{v_i\})$ denote the optimal schedule for the data transmission along the path from $v_i$ to $v_0$ (or BS). Consider an optimal schedule $U(\{v_i\})$ for the path $P_{d_0}$ from $v_i$ to $v_0$, where path $P_{d_0} = \{v_1 = v_i^0, v_i^1, \ldots, v_i^n, v_0\}$ and any two nodes on the path satisfy the sequential relation. Let $t_j$ denote the required data relaying time for $v_i^j \in P_{d_0}$. Let notation $[r_i^j]$ denote the schedule of node $v_i$. Let $t_{k+1}$ denote the total transmission time required for transmitting data from $v_i$ to $v_0$. We obtain $t_{k+1} = t_1 + t_2 + \ldots + t_k$. The optimal schedule $U(\{v_i\})$ of node $v_i$ can be represented by the structure, as follows:

$$U(\{v_i\}) = \left[ v_i^0 \right] \rightarrow \left[ v_i^1 \right] \rightarrow \ldots \rightarrow \left[ v_i^n \right] = \{v_i\} \quad 1 \leq i \leq n,$$

$$v_i^0, \ldots, v_i^n \in P_{d_0} = \{v_i\}$$

(1)

Fig. 9 shows an example to further illustrate the sequential relation and optimal schedule $U(\{v_1\})$. In this example, assume that the amount of data of node $v_1$ is 8 units. The data of node $v_1$ would be transmitted to $v_0$ along the path $P_{d_0} = \{v_1, v_2, v_3, v_4, v_0\}$. Assume that the rates $r_0$ and $r_{0,0}$ are 8 and 4 units/slot, respectively. Let $t_1$ and $t_2$ denote the required transmission times of $v_1$ and $v_3$, respectively. Therefore, $t_1$ and $t_2$ require 1(8/8) and 2(8/4) slots, respectively. Because the time slots allocated to $v_1$ and $v_3$ cannot be overlapped, the total transmission time is $t_{k+1} = t_1 + t_2 = 3$. Notation $\{v_i\} \rightarrow \{v_n\}$ indicates that the transmissions
for nodes \( v_i \) and \( v_o \) have a sequential relation. Hence, the optimal schedule of node \( v_i \) can be represented by \( U(v_i) = \{v_i^{t_1}\} \rightarrow \{v_i^{t_2}\} \rightarrow \{v_i^{t_3}\} \rightarrow \cdots \). After the node \( v_i \) has been scheduled, it is called a scheduled node.

More than one data flow may be requested for transmitting data to the BS. However, scheduling two or more data flows is more complex than scheduling only one data flow. This section describes the manner in which the two data flows are scheduled and represented by considering the three relations. A data structure of a Scheduled Node Group (SNG) is used to maintain the set of all the scheduled nodes. The elements in a SNG can be extended from a single source node to multiple source nodes. Let \( U(SNG) \) denote the optimal schedule of all nodes in the SNG. When the number of elements in a SNG is more than one, the nodes in the SNG may have nonparallel, parallel, and sequential relations. For simplicity, we discussed the nonparallel and parallel relations and ignored the sequential relation because the sequential relation has been discussed in Exp. (1). It is worthy to note that a sequential relation occurs if the distance between the source node and the BS is more than one hop. We first discuss the nonparallel relation. Consider two paths, \( P_{x0} \) and \( P_{y0} \). Let the \( x \)th node \( v_i^x \) on path \( P_{x0} \) and the \( y \)th node \( v_i^y \) on path \( P_{y0} \) have a nonparallel relation, and their required transmission times are \( t_i \) and \( t_j \), respectively. The optimal schedule of these two nodes is represented by

\[
\left\{ \{v_i^{t_1}\} \cup \{v_j^{t_2}\} \right\}^k
\]

where the total required transmission time \( t_i + t_j \). If \( v_i^o \) and \( v_j^o \) have a parallel relation, the time slots allocated to \( v_i^o \) and \( v_j^o \) can be overlapped for \( t_k \) slots, where \( t_k = \min(t_i, t_j) \). Therefore, the optimal schedule of nodes \( v_i^o \) and \( v_j^o \) can be represented by

\[
\left\{ \{v_i^{t_1}\} \cup \{v_i^{t_2}\} \right\}^k \cup \{v_j^{t_1}\}^{t_k}
\]

where \( v \) is the node that requires longer transmission time in the parallel relation and \( t_k \) is extra required transmission time for \( v \) after the parallel transmission.

Because the relations among the nodes in the SNG can be complex, the following section discusses the derivation of an optimal scheduling \( U(SNG) \) in a systematic manner. In the first step, we discussed the sequential relation and classified all nodes in the SNG into \( k \) sets \( S_1, S_2, \ldots, S_k \), where any two sets in \( S_1, S_2, \ldots, S_k \) satisfy the sequential relation. Based on the \( k \) classified sets, the optimal schedule \( U(SNG) \) can be represented by the sequential transmission of the \( k \) sets, as shown in (4), where \( opt(S_i) \) denotes the optimal transmission schedule of nodes in set \( S_i \).

\[
U(S_{NG}) = \{opt(S_1^0) \rightarrow opt(S_1^1) \rightarrow \cdots \rightarrow opt(S_1^p)\}^2
\]

In the following, we further discuss the nonparallel relation between nodes in a set \( S_i \). In the optimal schedule of \( S_i \), the transmission schedules in \( S_i \) can be further divided into \( l \) sets \( N_1, N_2, \ldots, N_l \), where any two sets in \( \{N_1, N_2, \ldots, N_l\} \) satisfy the nonparallel relation, as shown in (5).

\[
opt(S_i) = \{N_{i1}^1 \# N_{i2}^2 \# \cdots \# N_{il}^{t_{i}}\}^2, i \in K
\]

In the optimal transmission schedule of \( N_j \), assume that \( m \) nodes are allowed for transmitting data simultaneously. The optimal schedule for each \( N_j \) can be represented as shown in (6), where \( t_{i+1} = \min \{i_1, i_2, \ldots, i_m\} \).

\[
N_j = \{p_{j1}^{i_1} \# p_{j2}^{i_2} \# \cdots \# p_{jm}^{i_m} \}^2, j \in K
\]

The following recursive equations, (7) and (8), implement the dynamic programming algorithm for calculating the optimal uplink schedule.

\[
S_{opt} = U(S_{NG}) = \arg \min_{S_{NG} \subseteq S_{NG}} \left\{U(S_{NG} \cup S_{NG}^0, S_{NG}^1) \right\}
\]

Let \( 2^{SNG} \) denote the power set of \( SNG \). The recursive equation (7) derives the optimal schedule of \( SNG \) from the solutions of two smaller sets, \( SNG^0 \) and \( SNG^1 \). Consider the example shown in Fig. 9. Let a scheduled group SNG be \( \{v_{in}, v_{jn}, v_{kn}\} \). The optimal solution \( U(SNG) \) can be derived based on the following three schedules: \( U(\{v_{in}\} \cup \{v_{jn}, v_{kn}\}) \), \( U(\{v_{jn}\} \cup \{v_{in}, v_{kn}\}) \), and \( U(\{v_{kn}\} \cup \{v_{jn}, v_{in}\}) \). The recursive relation of (7) can be reapplied to derive the optimal solution of \( U(\{v_{jn}\} \cup \{v_{kn}\}) \) and \( U(\{v_{kn}\} \cup \{v_{jn}\}) \). Each of these optimal solutions of \( U(\{v_{jn}\} \cup \{v_{kn}\}) \) and \( U(\{v_{kn}\} \cup \{v_{jn}\}) \), which are optimal schedules for the transmission from the single node to the BS.

Let symbol \( \oplus \) denote the basic operation for deriving the optimal schedule. Equation (8) copes with the problem of using two optimal schedules, \( U(SNG_1) \) and \( U(SNG_2) \), to derive an optimal schedule for node set \( SNG_1 \cup SNG_2 \).

\[
U(S_{NG} \cup S_{NG}) = U(S_{NG} \oplus S_{NG}) = U(S_{NG} \oplus S_{NG})
\]

After the deriving process, the SADD can combine the optimal transmission schedules of two smaller node sets, \( SNG_1 \) and \( SNG_2 \), and derive the optimal schedule for a larger set, \( SNG_1 \cup SNG_2 \). The following further discusses the details of operation \( \oplus \). Consider the following optimal schedules.

\[
U(S_{NG} \oplus S_{NG}) = \{opt(S_{NG}^0) \rightarrow opt(S_{NG}^1) \rightarrow \cdots \rightarrow opt(S_{NG}^p)\}
\]

To construct the optimal solution of \( U(S_{NG} \cup S_{NG}) \), the parallelization task and sequencing task must be executed on sets \( SNG_1 \) and \( SNG_2 \). The parallelization task aims to extremely exploit the parallel transmissions between sets \( SNG_1 \) and \( SNG_2 \) whereas the sequencing task aims to guarantee that the original sequential relations in \( SNG_1 \) and
$SNG_j$ are also valid in the optimal schedule of $U(SNG_i \cup SNG_j)$. When executing the parallelization task, transmission schedule $N_i \parallel N_j$ must be extremely exploited for any $N_i \in S_i$, and $N_j \in S_j$, under the following criteria.

$$CM \{P_i, P_j\} = 0, \forall P_i \in N_i, and \forall P_j \in N_j$$

Let $N_i(t)$ denote the transmission time of $N_i$. An exploitation of parallel transmission $N_i \parallel N_j$ can reduce the total transmission time from $N_i(t)+N_j(t)$ to $\min\{N_i(t), N_j(t)\}$. Because it has $3 \leq N_i \leq N_j$, the merged schedule, the slots allocated to $N_j$ must be arranged before the slots allocated to $N_i$.

Based on the tree shown in Fig. 1, an example is provided to illustrate the merging procedure of individual schedules of nodes $v_1$ and $v_5$. Based on the tree topology, the quality of each link and the bandwidth request of each node as shown in Fig. 1, the BS can obtain the optimal schedule of each node of $v_1$ and $v_5$.

$$U((v_1)) = \{v_1^{(2)}\}$$

$$U((v_5)) = \{v_5^{(2)}\}$$

$$U((v_1 + v_5)) = \{v_1^{(2)}, v_5^{(2)}\}.$$  

The optimal schedule of $U((v_1))$ contains one subsequence $[v_1^{(2)}]$, whereas the optimal schedule of $U((v_5))$ comprises four subsequences, $[v_5^{(2)}], [v_5^{(3)}], [v_5^{(4)}], and [v_5^{(5)}]$, which have the sequential relation. Notice that the sequential relation must be maintained in a merged schedule. To exploit the opportunities of a parallel relation, the BS examines whether the subsequence $[v_1^{(2)}]$ of $U((v_1))$ and each of the subsequences of $U((v_5))$ can be allocated at the same time slots. Because subsequences $[v_5^{(3)}]$ and $[v_5^{(4)}]$ satisfy the parallel relation, the merged schedule can be recorded as $[v_1^{(2)} \parallel v_5^{(3)}]$, the total transmission of which is 8 slots. This result is shown in Fig. 10.

![Fig. 10](image)

Fig. 10. An example of the merged schedule $[v_1^{(2)} \parallel v_5^{(3)}]$. Because the transmission time of subsequence $[v_1^{(3)}]$ is larger than that of $[v_1^{(2)}]$, $[v_1^{(3)}]$ can be further examined to merge more subsequences with a parallel relation. The five null slots shown in Fig. 10 represent the capacity for merging more subsequences.

As shown in Fig. 11, three of the five null slots can be allocated to subsequence $[v_1^{(3)}]$ because it has a parallel relation with $[v_5^{(3)} \parallel v_5^{(4)}]$. As shown in Fig. 11, the new merged schedule is $[v_1^{(2)} \parallel v_5^{(3)} \parallel v_5^{(4)}]$. Fig. 11. An example of the merged schedule $[v_1^{(2)} \parallel v_5^{(3)} \parallel v_5^{(4)}]$.  

Fig. 11 shows two null slots that can be assigned for the subsequence $[v_1^{(4)}]$, because it has a parallel relation with $[v_1^{(3)}]$. Fig. 12 shows the overall merging result, which can be represented by $[v_1^{(2)} \parallel v_5^{(3)} \parallel v_5^{(4)} \parallel v_5^{(5)}]$, because node $v_5$ has parallel relation with $v_1$.

$$u((v_1, v_5)) = \{v_1^{(2)} \parallel v_5^{(3)} \parallel v_5^{(4)} \parallel v_5^{(5)}\}.$$  

The total transmission time is 12 slots.

![Fig. 13](image)

Fig. 13. An example of optimal schedule $U(v_1, v_5)$ which can be obtained by merging $U((v_1))$ and $U((v_5))$.  

Fig. 14 shows the detailed steps of the proposed SADP algorithm. In Step 3, the optimal schedule for the data transmission from a single node $v_i$ to the BS along the path embedded in the scheduling tree is constructed and stored in the array $d[i][v_i]$. Based on (8) and the information stored in $d[i][v_i]$, the optimal transmission schedule of any two nodes $v_i$ and $v_j$ can be calculated and stored in $d[i][v_j]$. The operation for constructing the optimal transmission schedule for a larger set can be calculated based on the optimal schedules of smaller sets that are previously calculated. Consequently, the optimal schedule of $SN\ G^*\ $containing $k$ nodes can be obtained using $A \otimes SN\ G^*\ A$.  

Received Date: 6/23/2012, Revised Date: 2/10/2013.
where $A$ is each subset of $SN G^r$ and the calculation results are stored in $D[k][SN G^r]$ as shown in Step 8. When the value of $k$ reaches $n$, the final optimal schedule of the mesh network can be obtained and stored in $D[n][SN G^r]$, as shown in Step 8.

Input : A graph $G(V, E)$, $V=\{v_1, v_2, ..., v_n\}$, $D=[d_1, ..., d_n]$, a scheduling tree $T(V, E')$ and a matrix CM

Output : An optimum scheduling $U(SNG)$

1. $i=0$, $j=0$, $k=0$

2. $D[1]...[n][subset of SNG] = \Phi$

3. for ($m=i$; $i<n$; $i++)$
   
   $D[i][\{v_i\}] = \{v_{i1}, v_{i2}, ..., v_{ik}\}$

4. end for

5. for ($k=2$; $k<=n$; $k++)$

6. for ($j=1$; $j<=k$; $j++)$

7. for ($j=1$; $j<=k$; $j++)$

8. $D[i][SN G^r] = min (D[j][\{v_i\}] \odot D[i-j][SN G^r - \{A\}])$

9. end for

10. end for

11. end for

12. return $D[n][SN G^r]$

Fig. 14. The proposed Scheduling Algorithm with Dynamic Programming Approach (SADP).

IV. HEURISTIC SCHEDULING ALGORITHM (HSA)

The SADP algorithm applies the dynamic programming approach and can attain an optimal transmission schedule. However, the computational complexity of SADP is high. This section introduces a heuristic scheduling algorithm, called HSA, which has lower complexity and can achieve similar performance to the SADP algorithm.

Several prior studies considered only the parallel transmission to enhance the network throughput. However, a large number of packets transmitted to the SS neighboring the BS is buffered, because the BS can only receive data from one of its neighbors at a time.

To reduce the total transmission time, the proposed HSA initially considers the flow with maximal length. Let $P_{max}$ denote the flow path with maximal length in the tree. The BS first schedules the closest link of $P_{max}$ by slot. If no data arrives or is generated from the node of this link, the link will not be assigned a slot because of a lack of a transmission request. In this case, the BS randomly chooses one link from other links closest the BS. Afterward, the BS subsequently schedules the farthest link of $P_{max}$ to start the flow of path $P_{max}$ as soon as possible. This policy aims to reduce the congestion near the BS and minimize the maximum transmission time. The other links belonging to $P_{max}$ are subsequently scheduled in an order from the nearest SS to the farthest SS. In the scheduling of each slot, the maximal number of possible links not belonging to $P_{max}$ must be assigned to maximize the number of parallel transmissions. To reduce the congestion phenomenon, the link with least distance to the BS is scheduled first.

Based on the tree topology shown in Fig. 1(a), Fig. 15 shows an example to illustrate the basic concept of HSA. Because the flow of $v_3$ has the longest path $P_{3,0} = \{v_5, v_4, v_3, v_2, v_0 (BS)\}$ in the tree, the link $(v_2, BS)$ closest to the BS is scheduled first. As shown in Fig. 15, Slot 1 is first allocated to link $(v_2, BS)$. The HSA subsequently allocates time slot 1 to link $(v_3, v_2)$, which is the last link of the longest path $P_{3,0}$. The next step is to schedule links that can be simultaneously transmitted with the scheduled links, $(v_2, BS)$ and $(v_3, v_2)$. In this step, other links cannot be scheduled because they interfere with the previously scheduled links. After scheduling Slot 1, the proposed HSA schedules the links of $P_{max}$ again. Because node $v_3$ has remaining data and path $P_{3,0}$ is the longest path compared with the paths of other flows, the links on the path $P_{3,0}$ are rescheduled based on the concept of the proposed HSA mechanism. Consequently, Slot 2 is assigned to links $(v_2, BS)$ and $(v_3, v_2)$.

Following this schedule, node $v_3$ transmits its entire data to node $v_5$ after Slot 3. Hence, the longest path changes from $P_{3,0}$ to $P_{1,0}$. For path $P_{1,0}$, link $(v_2, BS)$ is the link nearest the BS. Therefore, the proposed HSA allocates Slot 4 to link $(v_2, BS)$. All other links cannot be scheduled because they interfere with link $(v_2, BS)$. For Slot 5, links on path $P_{1,0}$ are rescheduled in advance. Although link $(v_2, BS)$ has the highest priority, no traffic can be transmitted through this link at Slot 5. Therefore, the BS chooses one link from links closest to the BS. Slot 5 is assigned to link $(v_1, BS)$. Subsequently, the BS allocates Slot 5 to link $(v_2, v_3)$, which is the farthest link of the longest path. The similar procedure repeats until all requirements are scheduled.

Fig. 16 shows the pseudo code of the proposed HSA. Let $L(v_i)$ denote the level of $v_i$ in the path. Steps 3-14 schedule the link closest to the BS, whereas Steps 15 and 18 schedule the last link belonging to the longest path $P_{max}$. Steps 19-27 schedule the links belonging to $P_{max}$ that can be simultaneously transmitted with the scheduled links. Steps 28-36 schedule the remaining links that can be simultaneously transmitted with the scheduled links. Steps 37-42 remove the well-scheduled flows and start another loop to repeatedly schedule the remaining links. The Check_S_relation($v_i$, $U(v_j)$) aims to examine whether node $v_i$ can be scheduled for forwarding the flow of $U(v_j)$. The
Check\_P\_relation(v_j) checks whether node v_j can be scheduled for parallel transmissions.

Table I shows a comparison of the existing scheduling algorithms [10][11][12][13][14] with the proposed SADP and HSA algorithms in terms of spatial reuse, various link rate, delay optimum, and time complexity. The proposed optimal approach considers spatial reuse, link rate, and delay simultaneously, which result in a superior schedule to that of existing algorithms. The proposed algorithm (SADP) shown in Fig. 14 mainly consists of three loops. The first and second loops are executed at most n times, where n is the number of SSs in the network. Therefore, the time complexity is \(O(n^2)\). For the third loop, the SADP calculates the transmission schedule of each SN \(G'\) which contains k nodes. Because the number of possible \(SN G'\) is \(c_k^2\), the complexity of loop 3 is \(2^k\). Consequently, the total time complexity of the proposed SADP is \(O(n^2+2^k)\).

The region size of the considered environment was set to 500x500 units. One BS was deployed at the center of the service region. The simulation applied two distributions to model the placement of SSs in the considered environment. The uniform distribution, which uniformly deploys SSs in the considered environment, was applied in the first model. To observe the interference effect on the performance of each compared scheduling algorithm, the congruencing distribution, which constrains all SSs, was deployed within the central area of 400x400 units in size in the second model. The number of SSs in the mesh network varied from 5 to 25. The BS and SSs were clock synchronized. The bandwidth request of each SS varied from 1 Mbps to 3 Mbps. In the simulation, the system performed the request update every 100 frames. All SSs had a common transmission range. In the physical layer, each node accessed the same channel and used the OFDM burst.
profiles, as shown in Table II.

Fig. 17(a) shows a scenario of the considered environment with uniform distribution. The x-axis and y-axis, ranging from 0 to 500 units, represent the coordinates system of the service region. Each unit represents 100 m. The location of the BS was set at the center of the region and marked by a black rectangle symbol. Twenty-five SSs were uniformly deployed in the area. The z-axis denotes the traffic requirement of each SS. As shown in Fig. 17(a), each SS applies a proper modulation based on the transmission distance and SNR value. Fig. 17(b) shows 25 SSs placed according to the congregating distribution. Compared with Fig. 17(a), the distances between two SSs and the SSs and BS reduced. Hence, SSs can apply superior modulation, as shown in Fig. 17(b).

Fig. 18 shows a comparison of the proposed SADP and HSA with the other three algorithms in terms of the average throughput. As shown in Fig. 18, the SSs are deployed with uniform distribution. The network throughput generally increases with the number of nodes when the network capacity can support the required bandwidth requests. When the bandwidth requests approach the upper bound of the network capacity, the system throughput increases slowly. The existing algorithms, including NNBF, FNBF, and NMRF may arrange for all children of the BS to receive data at the same time slot, resulting in a situation in which the BS is unable to receive any data from its children at that slot. The proposed SADP applies dynamic programming and can avoid this situation. It arranges for one child to send data to the BS and a few other children to receive data at the same time slot. This reduces the number of time slots required by SADP, which increases the network throughput. Consequently, the proposed SADP outperformed the other four algorithms regarding network throughput in all cases. In addition, the performance result of the proposed HSA is closer to that of the proposed SADP. This occurred because the proposed HSA exploits more opportunities for parallel transmissions, which reduces the transmission delay for flows that have a larger hop count to the BS. In general, the proposed SADP and HSA outperformed the other three algorithms in terms of network throughput.

Instead of applying uniform distribution, Fig. 19 shows the application of the congregating distribution to determine the location of each SS. As shown in Fig. 17(b), the SSs are deployed closer to the BS compared with the deployment of SSs in Fig. 17(a). This also indicates that the SSs in Fig. 19 have superior modulation and a higher transmission rate. Consequently, Fig. 19 shows superior performance to Fig. 18 in terms of network throughput. In addition, the proposed SADP exhibited superior performance to the other four schemes when the number of nodes increased to 25. Generally, the proposed SADP and HSA algorithms outperformed the other three algorithms under congregating distribution.

Fig. 20 shows a comparison of the proposed SADP and HSA algorithms with the existing three algorithms in terms of the peak throughput. In the simulation, the number of nodes ranged from 5 to 25. The traffic of each node was randomly selected as 1 Mbps, 2 Mbps, or 3 Mbps. As shown in Fig. 20, all curves generally grow when the number of nodes increases. The NMRF algorithm outperforms the other four scheduling algorithms because it always schedules the link with a higher data rate in advance. Although the NMRF algorithm achieves maximal peak throughput at a few slots, the links with lower data rates may interfere with each other and restrict the degree of parallel transmissions in the remaining time slots. The proposed SADP exploits the opportunities of parallel transmissions and arranges more possible transmissions simultaneously. Consequently, the peak traffic of SADP
approaches that of FNBF. The proposed HSA exhibited superior performance to the NNBF and FNBF algorithms in terms of the peak throughput. This occurs because the proposed HSA schedules the links with the smallest and largest hop counts in advance. This alleviates the congestion phenomenon that occurs near the BS. Fig. 21 shows the application of the congregating distribution to the node placement. The proposed SADP and HSA outperformed the NNBF and FNBF with improvement of 15%-20% in terms of the peak traffic.

![Fig. 20. The comparison of five compared algorithms in terms of the peak traffic by applying uniform distribution to place SSs.](image1)

Fig. 21. The comparison of five compared algorithms in terms of the peak traffic by applying congregating distribution as deployment policy.

Fig. 22 shows the varying 1-hop traffic ratio, which refers to the ratio of traffic initiated from the one-hop neighbors of the BS to the traffic initiated from all nodes. The proposed SADP and HSA algorithms were compared with the three existing algorithms in terms of normalized network throughput. The performance results were normalized to ensure that the proposed SADP algorithm maintains a constant value of 1. The 1-hop traffic ratio ranged from 30% to 100%. The number of nodes was 25. As shown is Fig. 22, the network throughput generally increases with the 1-hop traffic ratio. This occurs because the nodes nearest to the BS can directly transmit data to the BS; thus, the network throughput can be easily increased without data forwarding. Consequently, the BS can continuously receive data from its neighbors when the 1-hop traffic ratio increases. In general, the proposed SADP outperforms all heuristic algorithms in terms of network throughput. This occurs because the proposed SADP can obtain the optimal scheduling even if the unbalanced traffic happened in the network. In addition, the performance of the proposed HSA outperforms the other three algorithms because the HSA schedules stations closest to the BS in advance, which reduces the occurrence of the congestion problem. In addition, the throughputs of the FNBF and NMRF algorithms increase rapidly when the 1-hop traffic ratio varied from 70 to 100. This occurred because of the considerable interference that often occurs at the two-hop neighbors of the transmitting node. Hence, the higher 1-hop traffic ratio can reduce numerous forwarding transmissions to avoid congestion.

![Fig. 22. The comparison of the proposed SADP and HSA and the other three schemes in terms of the normalized throughput by varying 1-hop traffic ratio under uniform distribution.](image2)

Fig. 23 shows a comparison of five scheduling algorithms by applying congregating distribution as the policy of node placement. Because the distances between SSs are reduced, the interference has a greater effect on the performance of network throughput. The proposed SADP exhibited superior performance to the other scheduling algorithms in the interference-rich environment. This occurred because the proposed SADP applies dynamic programming to obtain the optimal solution. Because the performance of the compared five algorithms were normalized based on the performance of the SADP, the normalized throughput of the proposed HSA and other existing algorithms, including FNBF, NNBF, and NMRF increased slowly.

![Fig. 23. The normalized throughput versus 1-hop traffic ratio by applying the congregating distribution as deployment policy.](image3)

Fig. 24 shows the performance of the five compared scheduling algorithms in terms of the average buffering time of arrived packets and the average throughput achieved by each level in the scheduling tree. The transmission requests of each node ranged from 1 Mbps to 3 Mbps. Let the packet generation rate denote the number of packets generated by each node per second. The packet generation rate of each node ranged from 10000 to 30000 packets per second (pps). The packet size was 100 bytes. A packet was dropped to avoid the buffer overflow problem when its buffering time was more than 250 ms. Let \( \zeta_{i}^{\text{max}} \) denote the amount of data forwarded by node \( v_i \). Let \( \zeta_{i}^{\text{max}} \) denote the amount of data generated from node \( v_i \). Let \( \zeta_{i} \) denote the total amount of data in the buffer of node \( v_i \). Exp. (9) evaluates the value of \( \zeta_{i} \).

\[
\zeta_{i} = \zeta_{i}^{\text{max}} + \zeta_{i}^{\text{max}} \tag{9}
\]
Let Boolean variable $\sigma_{i,k}^{\text{in}}$ indicate whether the $k$-th bit in the buffer of node $v_i$ is transmitted successfully; that is,

$$
\sigma_{i,k}^{\text{in}} = \begin{cases} 
1, & \text{if the } k \text{-th bit in the buffer of node } i \text{ is transmitted successfully;}
\smallskip 
0, & \text{otherwise.}
\end{cases}
$$

(10)

Let Boolean variable $\sigma_{i,k}^{\text{out}}$ indicate whether the $k$-th bit in the buffer of node $v_i$ is dropped; that is

$$
\sigma_{i,k}^{\text{out}} = \begin{cases} 
1, & \text{if the } k \text{-th bit in the buffer of node } i \text{ is dropped;}
\smallskip 
0, & \text{otherwise.}
\end{cases}
$$

(11)

Let $t_{i,k}^{\text{in}}$ and $t_{i,k}^{\text{out}}$ denote the arrival time and departure time of the $k$-th bit of the node $v_i$, respectively. Let $\tau_{i,k}^{\text{drop}}$ denote the dropping time of the $k$-th bit of the node $v_i$. Let $T_{\text{height}}$ and $L_h$ denote the height of the scheduling tree and the set of nodes on the $h$-th level of the scheduling tree, respectively. Let $\omega_h$ denote the average buffering time of level $h$. Exp. (12) shows the derivation of $\omega_h$.

$$
\omega_h = \frac{\sum_{i \in L_h} \sum_{k=1}^{2^\eta} (t_{i,k}^{\text{out}} - t_{i,k}^{\text{in}}) \times \sigma_{i,k}^{\text{in}} + \sum_{i \in L_h} \sum_{k=1}^{2^\eta} (\tau_{i,k}^{\text{drop}} - t_{i,k}^{\text{in}}) \times \sigma_{i,k}^{\text{out}}}{\sum_{i \in L_h} \sum_{k=1}^{2^\eta} \varepsilon_i \cdot \sigma_{i,k}^{\text{in}}}, \quad 1 \leq h \leq T_{\text{height}}
$$

(12)

As shown in Fig. 24, the value of $T_{\text{height}}$ was set at 5. In general, the buffering time $\omega_h$ is larger than $\omega_h, \quad 2 \leq i \leq 5$. This occurred because the BS is the common destination of all data flows; however, the BS cannot receive multiple data from various nodes at the same slot. Therefore, the data are buffered for a long time at the nodes in $L_1$. Fig. 24 shows that the proposed SADP outperforms the other four scheduling algorithms in terms of average buffering time. This occurs because the proposed SADP arranges each node to transmit data in suitable time slots and requires the least transmission time to complete all traffic requirements. In addition, the average buffering times $\omega_1, \omega_3$, and $\omega_5$ of the proposed HSA are closer to the proposed SADP algorithm than those of the NNBF and NMRF algorithms. This occurs because the proposed HSA schedules the links with the smallest and largest hop counts in advance to alleviate the traffic congestion that occurs near the BS and reduce the transmission delay. Consequently, the proposed HSA has smaller buffer delay at the nodes in $L_3$ than the other algorithms. The FNBF schedules the node with the largest path length in advance, which reduces the average buffer delay of the nodes in $L_3$. However, the transmissions of the nodes in $L_4$ are blocked, resulting in a large value of $\omega_L$.

Fig. 24 also shows a comparison of five algorithms in terms of the average throughput achieved by each tree level. The average throughput achieved by each level was measured every second. As shown in Fig. 24, all data can be transmitted rapidly by applying the proposed SADP. Furthermore, because the proposed HSA prior allocates the time slot for the nodes on the first and fifth levels, the first and fifth levels have superior throughput than the other levels. The large amount of throughput achieved on the first level indicates that the data can be successfully transmitted to the BS.

It is worth to notice that the traffic congestion occurs on level $i$ when the amount of data of level $i+1$ is larger than that of level $i$. As shown in Fig. 24, the FNBF algorithm results in traffic congestion on the first level, thereby reducing the amount of data received by the BS. The NNBF encounters the same problem on the third level. In general, the proposed HSA exhibits superior performance than the other algorithms in terms of average throughput achieved on each level.

Let $\text{MITT}$ delay denote the maximal tolerable transmission delay for a data flow from a source node to the BS. Fig. 25 shows the effect of $\text{MITT}$ delay on the packet dropping ratio. The $\text{MITT}$ delay ranged from 1 ms to 500 ms. The packet dropping ratio generally decreases with the $\text{MITT}$ delay. The proposed SADP initially schedules the packet transmission flow by flow. Based on each flow schedule, the SADP exploits maximal parallelism and avoids interference between neighboring nodes. Therefore, all flows can be scheduled with minimal uplink transmission time. Consequently, the proposed SADP algorithm outperforms the other four scheduling algorithms. In addition, the proposed HSA algorithm exhibits superior performance than the FNBF and NMRF algorithms because it can reduce transmission delay.

Fig. 24. The comparison of five algorithms in terms of the average throughput achieved on each level and the average buffering time by applying uniform distribution as the deployment policy.

Let $\text{MITT}$ delay denote the maximal tolerable transmission delay for a data flow from a source node to the BS. Fig. 25 shows the effect of $\text{MITT}$ delay on the packet dropping ratio. The $\text{MITT}$ delay ranged from 1 ms to 500 ms. The packet dropping ratio generally decreases with the $\text{MITT}$ delay. The proposed SADP initially schedules the packet transmission flow by flow. Based on each flow schedule, the SADP exploits maximal parallelism and avoids interference between neighboring nodes. Therefore, all flows can be scheduled with minimal uplink transmission time. Consequently, the proposed SADP algorithm outperforms the other four scheduling algorithms. In addition, the proposed HSA algorithm exhibits superior performance than the FNBF and NMRF algorithms because it can reduce transmission delay.

Fig. 25. The packet dropping ratio by varying $\text{MITT}$ delay.
Fig. 26 shows the effect of a non-uniform transmission rate on the network throughput. In the simulation, 100 nodes were uniformly deployed within an area with 500 × 500 units. The bandwidth request of each node was randomly generated and the value of the bandwidth request ranged from 0 Mbps to 2 Mbps. The experiment was conducted for 2 hours. To vary the transmission rate of each link, the signal was corrupted by adding Gaussian noise for 10 minutes at 30, 60, and 90 minutes. As shown in Fig. 26, all curves were generally stable at the beginning. However, the performance of all scheduling algorithms substantially reduced after 30 minutes. This occurred because considerable background noise reduces the transmission success ratio. Therefore, SSS change to lower coding rates, resulting in inferior throughputs. In general, the proposed HSA outperforms NMRF, NNBF, and FNBF regarding of network throughput. This occurs because the proposed HSA algorithm arranges more possible transmissions simultaneously and alleviates the traffic congestion phenomenon.

Fig. 26. The comparison of four algorithms in terms of the network throughput. The transmission rate of each link is varied because the signal is corrupted by adding Gaussian noise for 10 minutes at the 30, 60 and 90 minutes.

VI. CONCLUSION

Several heuristic approaches have been proposed to manage the scheduling problem in WiMAX mesh networks. However, their performances highly depend on the network topology and the bandwidth request of each node and they do not achieve optimal performance in all cases. This paper proposes the SADP algorithm for minimizing the number of time slots required for a given set of bandwidth requests in WiMAX mesh networks. By considering sequential, nonparallel, and parallel relations, a valid and optimal schedule was constructed using a dynamic programming strategy, which avoids redundant computations. This paper also proposes a heuristic algorithm, called HSA, to reduce the computing complexity. The proposed HSA achieves similar performance to the SADP algorithm. The simulation results indicated that the proposed SADP and HSA algorithms outperform the existing algorithms in terms of average network throughput, peak traffic, and packet dropping ratio.

VII. REFERENCES