Fair Scheduling and Throughput Maximization for IEEE 802.16 Mesh Mode Broadband Wireless Access Networks

Muhammad Mahbub ALAM, Md. Abdul HAMID, Md. Abdur RAZZAQUE, Nonmembers, and Choong Seon HONG, Member

SUMMARY Broadband wireless access networks are promising technology for providing better end user services. For such networks, designing a scheduling algorithm that fairly allocates the available bandwidth to the end users and maximizes the overall network throughput is a challenging task. In this paper, we develop a centralized fair scheduling algorithm for IEEE 802.16 mesh networks that exploits the spatio-temporal bandwidth reuse to further enhance the network throughput. The proposed mechanism reduces the length of a transmission round by increasing the number of non-contending links that can be scheduled simultaneously. We also propose a greedy algorithm that runs in polynomial time. Performance of the proposed algorithms is evaluated by extensive simulations. Results show that our algorithms achieve higher throughput than that of the existing ones and reduce the computational complexity.

Key words: wireless mesh networks, broadband wireless access, fair scheduling, quality of service, bandwidth reuse

1. Introduction

To extend the service quality of the last-mile wireless access networks, IEEE 802.16 broadband wireless access (BWA) networks have brought a new dimension that provides higher coverage and data rates. It has two modes of operation: the point-to-multipoint (PMP) and the optional mesh mode. In the PMP mode, all subscriber stations (SSs) have to be within the transmission range and clear line-of-sight (LOS) of the base station (BS). On the other hand, in the mesh mode, stations form a multihop network and act as router to forward packets of its neighbor stations. In this case, there is no need to have a direct link from SS to BS, i.e., packets are traveled in a multi-hop fashion. In this paper, we concentrate on the problem of fair scheduling in mesh mode, since it brings down additional challenges compared to the PMP mode.

The mesh mode supports two types of scheduling: centralized scheduling and distributed scheduling. Centralized scheduling works similar to the scheduling of PMP mode with the exception that the SSs form a mesh network. It is a combination of both the centralized scheduling of the PMP mode and the multihop and non-line-of-sight (NLOS) features of the mesh mode; and is the topic of this paper.

In a BWA mesh network, data of all the SSs pass through the BS to the Internet. Since all the SSs use the same BS, it is intuitive that the throughput of different connections can vary depending on the location of the SSs to whom they are connected. Moreover, as shown in [1], if a particular SS is more than two-hop away from the BS, then a connection of this SS might not send or receive traffic at all (i.e., the connection may starve); which is not only unfair but undesirable as well. On the other hand, multi-hop wireless networks can increase the bandwidth utilization using the same channel in different parts of the network [2]. Therefore, the main challenges of scheduling in a mesh network are two folds: (1) to ensure that every connection gets equal access from the network, irrespective of their locations, and (2) to achieve optimum bandwidth utilization to increase the overall network throughput.

In this paper, we develop a fair scheduling scheme to meet the above challenges. We propose a spatio-temporal bandwidth reuse method (explained in Sect. 4.2), that achieves better throughput than the spatial reuse method discussed in [2] and [3]. In [3], a clique construction algorithm is proposed to ensure per-client fairness and to maximize the throughput by exploiting the spatial bandwidth reuse proposed in [2]. Our works differ from [3] in that: (1) we use a complete r-partite graph as a unit of scheduling instead of a clique, which achieves better throughput (Sect. 4.2), (2) we propose a localized algorithm to reduce the complexity, which partitions the bandwidth reuse graph into smaller subgraphs and maximizes the bandwidth reuse locally (Sect. 5.2), and (3) finally, we present a greedy algorithm, which considers the complete r-partite graph as a unit of scheduling and has a polynomial time complexity (Sect. 5.3).

This paper is organized as follows. Section 2 describes the related works and Sect. 3 describes the system model and assumptions. The basic principle of the proposed scheduling algorithms is explained in Sect. 4. Section 5 explains the proposed scheduling algorithms and in Sect. 6, we extend the proposed scheduling algorithms to adopt the traffic classes of IEEE 802.16. Performance evaluation of the proposed algorithms is presented in Sect. 7 and Sect. 8 concludes the paper.
2. Related Works

A general discussion about wireless mesh networks (WMNs) is presented in [4]. This paper surveyed the state-of-the-art technologies and mechanisms of WMNs. Kuran and Tugcu presented a survey on emerging BWA technologies in [5]. In [2], Nelson and Kleinrock defined a new channel access protocol called S-TDMA, extending the traditional TDMA protocol, which operates in multihop packet radio networks with fixed nodes and uses collision-free centralized scheduling. The authors also presented an approximate solution to determine the channel assignment capacity for the links of the networks to minimize the average delay.

In [3], Salem and Hubaux proposed a centralized scheduling solution for wireless mesh networks. They presented a clique construction algorithm, which maximizes the throughput by capitalizing the spatial reuse of bandwidth. However, the complexity of their scheduling algorithm depends on clique enumeration, which is proven to be NP-hard.

In [6], a max-min fair bandwidth allocation algorithm is presented. The authors studied the association control problem and consider bandwidth constraints of both the wireless and backhaul links. Their formulation of the problem indicates strong correlation between fairness and load balancing, which allows to use a load balancing technique to obtain an optimal max-min fair bandwidth allocation. Since this problem is NP-hard, they presented algorithms to achieve a constant-factor approximate max-min fair bandwidth allocation.

In [1], the authors studied per-TAP fairness and end-to-end performance in WMNs (multi-hop wireless backhaul networks). They proposed an inter-TAP fairness algorithm that aims to achieve the per-TAP fairness without modifying the TCP protocol.

Performances of the scheduling in IEEE 802.16 based wireless mesh networks are evaluated in [7] and [8]. In [9], a fair multihop scheduling algorithm for IEEE 802.16 mesh networks is proposed. The proposed algorithm determines the transmission order of the nodes and provides equal bandwidth to each node. Since node based fairness is proposed; if multiple clients are connected with the SSs, fairness might be violated. In [10], a fair scheduling algorithm is proposed for IEEE 802.16 mesh networks in distributed scheduling mode.

3. System Model and Assumptions

3.1 Brief Introduction of IEEE 802.16 Mesh Mode

IEEE 802.16 [11] provides fixed broadband wireless access with QoS guarantee and supports two operational modes: Point-to-multipoint and the optional mesh mode. The wireless mesh network adopts the Time Division Multiple Access (TDMA) method between SSs and between SSs and BS, where the channel is divided into slots using time sharing and fixed number of slots are grouped together to form a frame. A frame is divided into a control subframe and a data subframe.

The mesh mode supports two different types of scheduling: centralized and distributed scheduling. In distributed scheduling, the SSs negotiate among themselves with a three-way handshaking procedure to select the slots for data transmission. Distributed scheduling is divided into two modes: coordinated and uncoordinated. In the coordinated mode, scheduling messages are exchanged in a collision-free manner; whereas, scheduling message can collide in the uncoordinated mode. In centralized scheduling, the BS acts as a scheduler and determines transmission and reception slots for each SS. The BS first collects bandwidth requests from all SSs, and then calculates and distributes the transmission and reception schedules to all SSs. However, in the centralized scheduling, if the hop count is very high, the delay of the packets might be high; especially, the best effort (BE) traffic will suffer a lot if the network favors the real-time and multimedia applications.

There are four distinct service classes defined in the IEEE 802.16 specification [11]. Unsolicited grant service (UGS) is designated for fixed-size data with periodic intervals. Real-time polling service (rtPS) is similar to UGS but for variable-rate traffic; such as, MPEG video data. Non-real-time polling service (nrtPS) is designed for applications that are not sensitive to delay and jitter. Finally, the Best Effort (BE) service has no specific demand.

3.2 System Description

We consider an IEEE 802.16 based network in mesh mode, which uses centralized scheduling. The mesh network is used as BWA network and one BS connects the network to the Internet. There are \( N \) SSs, denoted by \( S_{S_i} \), where \( i = 1, 2, \ldots, N \). These SSs form the mesh network. The clients (users) are connected with the SSs using the existing LAN technologies; for example, 802.11 (WiFi). Each SS creates one or more connections to deliver and receive data to/from the BS. Let \( C_j \) denote the number of connections in \( S_{S_i} \) and the \( j \)-th connection of \( S_{S_i} \) is denoted by \( C_{j'} \), where \( j = 1, 2, \ldots, C_j \). We denote the set of SSs by \( \mathbb{N} \) and the set of the connections of \( S_{S_i} \) by \( \mathbb{C}_i \). SSs that are not directly connected with the BS, use some intermediate SSs to send and receive their data. Furthermore, the access network may provide QoS guarantee to the clients and support different traffic classes as specified in the standard. An example of IEEE 802.16 based wireless mesh access network is shown in Fig. 1.

The mesh network can be represented as a directed graph, where the BS and SSs are the vertices of the graph. A link exists between two SSs or between one SS and the BS, if they are within the communication range of each other. We denote a link by \( L_{x,y} \) or \( L_{y,z} \), if it is from \( S_{S_x} \) to \( S_{S_y} \) or from \( S_{S_y} \) to \( S_{S_z} \), respectively. We define a link as upstream link, if it carries traffic from the SSs toward the BS or down-
stream link, if it carries traffic from BS toward the SSs. Note that if \( x = 0 \) or \( y = 0 \) in \( l_{x,y} \), then it is the downlink from BS to \( SS_y \) or uplink from \( SS_x \) to BS, respectively. Also, there exist some links, which are not used to carry traffic but activated unintentionally and are known as interference link [3] (for example, in Fig. 1, the links shown by dotted lines are interference links). We denote the set of upstream links by \( U \) and the set of downstream links by \( D \). Also, a link is active in the period when it carries traffic.

All traffic from the SSs to the BS are termed as uplink traffic and the traffic from a particular connection of a particular SS toward the BS is defined as uplink flow. Hence, a particular connection might aggregate the traffic of multiple applications from a single client those pass through the BS and are delivered to the Internet. Similarly, all traffic from the BS to the SSs are termed as downlink traffic and the traffic from the BS to a particular connection of a particular SS is defined as downlink flow. Also, note that the mesh network is a multihop broadband access network, so a connection might either be single hop or multihop depending on the hop count of the SS from the BS.

3.3 Assumptions

We assume that the SSs are static; so, the topology of the access network does not change frequently and it changes only when an SS joins or leaves the network. We assume that all the communication links are homogeneous and the channel is error free. The topology of the mesh network is known to the BS and the SSs. Also, the BS knows the traffic class and the current bandwidth demand of each connection of each SS. The scheduling decision is made by the BS and is delivered to all SSs.

Scheduling is done separately for uplink and downlink flows. Since the scheduling mechanisms are similar for both the uplink and downlink flows, a single scheduling mechanism is applicable to both. Therefore, in this paper, we only explain the mechanism for the uplink flows.

We assume that the network is used to connect clients to the Internet. So, the traffic only flows from the SSs to the BS and from the BS to the SSs. We also assume that the upstream traffic, downstream traffic and control messages are sent using different channels.

4. Fairness and Throughput in Mesh Networks

4.1 Problem Definition

The fairness of a scheduling mechanism is measured based on the throughput achieved by the active flows\(^1\) in a certain interval. In this paper, we aim at providing fair access to all the active flows (in terms of throughput) regardless of their spatial bias.

**Definition 1. Throughput of a Flow** — In uplink scheduling, the throughput of a flow (or a connection) is defined as the number of bytes received by the BS from that flow per unit time. Similarly, in downlink scheduling, the throughput of a flow is defined as the number of bytes received by the flow from the BS (i.e., the number of bytes forwarded by the BS for that flow) per unit time.

**Definition 2. Fair Scheduling** — A scheduling mechanism (either uplink or downlink) is defined as fair, if the throughput achieved by all the flows are equal in all time interval \([t_1, t_2]\) and is given by

\[
W_i(t_1, t_2) \geq W_{q,i}(t_1, t_2) \quad \forall S_i, \quad S \subseteq S, \quad p \in \mathbb{N}
\]

where, \(W_i(t_1, t_2)\) and \(W_{q,i}(t_1, t_2)\) are the throughput achieved in the interval \([t_1, t_2]\) by the flows \(F_i\) and \(F_{q,i}\), respectively. The inequality in Eq. (1) comes when \(F_{q,i}\) is partially backlogged and \(F_i\) is fully backlogged in the interval \([t_1, t_2]\).

**Definition 3. QoS-Aware Fair Scheduling** — In QoS-aware fair scheduling, there exist some flows having specific demands from the network. A scheduling mechanism is both QoS-aware and fair, if it first fulfills the requirements of the QoS-aware flows, and then, fairly distributes the remaining resources (for example, bandwidth) among the normal flows, which satisfies Eq. (1).

For simplicity of description, we first consider that no QoS-aware flow is present in the network and spatial reuse of frequency is not used. The scheduler assigns fixed number of slots to a link for a flow. Suppose, the hop count of \(F_i\) is one and the BS allocates \(S\) slots in the interval \([t_1, t_2]\) for the only link that carries the data of \(F_i\). If \(b\) bytes of data can be sent in a single slot, the throughput achieved by \(F_i\) is given by

\[
W_i(t_1, t_2) = \frac{S \times b}{t_2 - t_1}
\]

That is, each connection, one-hop away from the BS, can send \((S \times b)\) bytes of data in the interval \([t_1, t_2]\). We define

\[^1\text{In this paper, we use the terms flow and connection interchangeably. The flow corresponding to the connection } C_j \text{ is denoted as } F_j.\]
this amount of data as one transmission unit.

Now, assume that \( r_i^j \) is the route of the flow \( F_i^j \) from \( SS_i \) to BS, and it requires \( h_i \) hops to reach the BS. If \( F_i^j \) is assigned \( S \) slots in the interval \([t_1, t_2]\), then it requires \( h_i \times (t_2 - t_1) \) time to send \((S \times b)\) bytes of data to the BS. This implies that the throughput of a flow depends on the location of the SS, to whom a particular flow is attached. This is the so-called location-dependent fairness problem of the wireless mesh networks [1]. To overcome the problem or to provide fairness, irrespective of the location of a flow/connection, the BS needs to receive equal amount of data from each flow in a certain interval. Therefore, the BS has to assign \( S \) slots to all the links in the route of a flow in the interval \([t_1, t_2]\) for that flow. This ensures that the BS will receive exactly \((S \times b)\) bytes from each active flow, in the interval \([t_1, t_2]\). Thus, the scheduling mechanism provides fairness.

We define the interval \([t_1, t_2]\) as a transmission round, if the BS receives exactly one transmission unit from each active connection. The duration of a round, denoted as \( t_r \), is given by

\[
t_r = t_2 - t_1 = \sum_{i=1}^{N} \sum_{j=1}^{C_i} h_i \times S \times T_d,
\]

where, \( T_d \) is the transmission time for a slot. On the other hand, the number of slots required for the link \( l_{x,y} \), in one transmission round, is given by

\[
L_{x,y} = \sum_{i=1}^{N} \sum_{j=1}^{C_i} I(r_i^j),
\]

where, \( L_{x,y} \) is the load of the link \( l_{x,y} \) and \( I(r_i^j) \) is an indicator function given by

\[
I(r_i^j) = \begin{cases} 
1 & \text{if } l_{x,y} \in r_i^j \\
0 & \text{otherwise}
\end{cases}
\]

**Definition 4. Load of a Link** — The load of a link is defined as the number of flows for which the link forwards data. In other words, \( L_{x,y} \) is the number of transmission units, \( l_{x,y} \) is assigned in a transmission round.

Therefore, the number of total transmission units required in a round for upstream (or, downstream) scheduling, denoted as \( T \), is the sum of the loads of the upstream (or, downstream) links and is given by

\[
T = T_d \sum_{l_{x,y} \in U} L_{x,y}.
\]

The throughput achieved by the flow \( F_i^j \) can be found from Eq. (2) and Eq. (6), which is

\[
W_i^j(t_r) = \frac{S \times b}{T \times T_d}.
\]

Then, the network throughput in a round, denoted as \( W(t_r) \), is calculated as

\[
W(t_r) = \sum_{i=1}^{N} \sum_{j=1}^{C_i} W_i^j(t_r) = \sum_{i=1}^{N} \sum_{j=1}^{C_i} \frac{S \times b}{T \times T_d}.
\]

Thus, the network throughput in a round is inversely proportional to \( T \). Therefore, the network throughput can be maximized, if we can minimize \( T \).

### 4.2 Throughput Maximization

The throughput of the network can be enhanced by reducing the number of transmission units in a transmission round. This is usually done by exploiting the spatial reuse of bandwidth, i.e., by scheduling multiple non-contending links simultaneously [2]. In this subsection, we design a fair scheduling mechanism that achieves more reuse of the bandwidth than that of the spatial reuse. The proposed mechanism uses spatio-temporal bandwidth reuse, which further reduces the length of a transmission round than that of the spatial bandwidth reuse and maximizes the network throughput.

The basic idea behind the scheduling mechanism is to map the network topology into a suitable matrix. The rows and columns of the matrix represent the active links of the network. The links form a symmetric binary matrix of size \(|U|\times|U|\) [2]. We define this matrix as Bandwidth Reuse Matrix (BRM), since it represents the links that can be scheduled simultaneously, and consequently, the links can reuse the bandwidth. The matrix is denoted as \( M \) and defined as \( M = [m_{r,c}] \), where \( m_{r,c} \) is the entries of the matrix and, \( r \) and \( c \) correspond to the \( r \)-th and \( c \)-th link in \( U \). The entries of the matrix are given by

\[
m_{r,c} = \begin{cases} 
1 & \text{if links } r \text{ and } c \text{ can be scheduled simultaneously} \\
0 & \text{otherwise},
\end{cases}
\]

where, \( 1 \leq r, c \leq |U| \). The bandwidth reuse matrix of the example mesh network shown in Fig. 1, for uplink scheduling, is given by

\[
M = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & 1 & 1 & 0
\end{pmatrix}
\]

where, the rows in \( M \) correspond to the links \((1,0), (6,0), (7,6), (2,1), (4,1), (3,2), (5,4), (8,7) \) and \((9,7)\), respectively. The bandwidth reuse matrix shown in Eq. (9) is constructed with the following assumptions

- An SS can only transmit or receive in the interval of a transmission unit.
- An SS can receive only from one neighbor in the interval of a transmission unit. So, all the other neighbors (except one transmitter) are kept silent.
• An SS can send to only one of its neighbors in the interval of a transmission unit.

The bandwidth reuse matrix can be represented as a graph [3]. We define the graph as Bandwidth Reuse Graph (BRG) and denote it as $G$. Each vertex in $G$ represents an active link in $U$. We denote the vertex corresponding the link $l_{x,y}$ as $v_{x,y}$. An edge between two vertices in $G$ (i.e., two links in $U$) denotes that the corresponding links are non-contending and can be scheduled simultaneously. Therefore, if a non-zero entry exists in $M$ for two links, the two corresponding vertices in $G$ are connected directly with an edge. For example, assume that $l_{x_1,y_1}$ and $l_{x_2,y_2}$ are the $r$-th and $c$-th link in $U$, respectively. Then, there is an edge between the vertices $v_{x_1,y_1}$ and $v_{x_2,y_2}$ in $G$, if $m_{x_c} = 1$ in $M$. The bandwidth reuse graph for the example network shown in Fig. 1 or the corresponding BRM in Eq. (9) is shown in Fig. 2.

Since the directly connected vertices in BRG are non-contending, the vertices that form a clique (i.e., vertices that are pairwise connected [12]) can be scheduled simultaneously. Each clique achieves a gain through spatial reuse of bandwidth. The gain of a clique is defined as the number of transmission units that the scheduling mechanism can save due to spatial reuse of bandwidth [3]. In [3], a fair scheduling mechanism is proposed, which maximizes the throughput of the network through spatial reuse of bandwidth. This mechanism schedules a set of cliques, which has the maximum gain subject to two conditions: i) the set of cliques includes each vertex in $G$ (i.e., each link in $U$) only once, and ii) the set of cliques includes all the vertices in $G$. The throughput achieved by the mechanism is higher than that of the TDMA mechanism.

However, the gain of a clique is maximum when the loads of all the links in the clique are equal. If the loads of the links in a clique are different, then bandwidth is not reused completely for all the transmission units allocated for that clique. In WMNs, links closer to the BS forward the traffic of the upstream nodes (downstream node for downlink scheduling) and have higher loads than a link at the edge of the network. This implies that the loads of the links in a mesh network can be different and it depends on the location of the links. Furthermore, two or more links can be scheduled simultaneously (i.e., reuse bandwidth), if they are distant enough, so that they do not interfere with each other, and thus, form a clique. As a result, the loads of the links in a clique are usually unequal for WMNs.

Therefore, we claim that only spatial reuse of bandwidth (and hence, scheduling the links as a set of cliques) is not optimum, even though it enhances the network throughput as compared to the TDMA mechanism. For example, if link $l_{x,y}$ is closer to the BS (say, $l_{x,y}$ connects one SS to the BS) and link $l_{x_1,y_1}$ is far away from the BS, then it is found that $L_{x,y} > L_{x_1,y_1}$. The clique construction algorithm creates a clique of cardinality 2 and the scheduler allocates $L_{x,y}$ transmission units for the clique. In such a scheduling, spatial reuse is achieved only for $L_{x_1,y_1}$ transmission units but the remaining transmission units (i.e., $L_{x,y} - L_{x_1,y_1}$) are used by only one link. Furthermore, there exist many other links far away from the BS (for example, $l_{x_2,y_2}$), those do not contend with $l_{x,y}$ but contend with $l_{x_1,y_1}$. It might also be true that not only $L_{x,y} > L_{x_1,y_1}$ but also $L_{x,y} + L_{x_2,y_2}$. That is, there exist many cliques of cardinality 2, those have a common vertex and the link corresponding to the common vertex is the mostly loaded link of the cliques. In such a scenario, it is possible to achieve a combination of both spatial and temporal reuse of bandwidth.

**Definition 5.** Spatio-Temporal Bandwidth Reuse — If $n$ links are individually non-contending with a particular link, $l_{x,y}$, but the links are pairwise contending (or, form an independent set in the BRG) and the combined load of the $n$ links are less than or equal to the load of $l_{x,y}$; then, a special scheduling is possible, where $n$ links are scheduled in different time with respect to each other and each of the $n$ links spatially reuses the bandwidth of $l_{x,y}$, achieving a combination of both spatial and temporal bandwidth reuse. We define this mechanism as Spatio-Temporal Bandwidth Reuse.

**Lemma 1.** Consider that there are $n$ cliques of cardinality 2 in the BRG, where each clique is an induced subgraph. Assume that there is a common vertex, $v_{x,y}$, in all cliques and the remaining vertices of the cliques are $v_{x_1,y_1}, v_{x_2,y_2}, \cdots, v_{x_n,y_n}$. Further, the link corresponding to the common vertex has the maximum load. Then, spatio-temporal bandwidth reuse can schedule the links of the cliques (i.e., $l_{x,y}$ and $l_{x_1,y_1}, l_{x_2,y_2}, \cdots, l_{x_n,y_n}$) simultaneously, those satisfy the following condition

$$L_{x,y} \geq L_{x_1,y_1} + L_{x_2,y_2} + \cdots + L_{x_n,y_n},$$

and maximizes the bandwidth reuse over individual scheduling of each clique.

**Proof.** Since, each of the $n$ links is non-contending with the
common link \( l_{x,y} \), the vertices corresponding to these links constitute \( n \) cliques of cardinality 2 with the common vertex, as shown in Fig. 3(a). As the links are mutually non-contending, a clique construction algorithm (for example, [3]) schedules the clique with the maximum gain and requires more transmission units for the remaining links. However, the links are individually non-contending with \( l_{x,y} \) and can be scheduled in different time interval with respect to each other, but within the interval of \( l_{x,y} \), as shown in Fig. 3(b). Thus, the spatio-temporal bandwidth reuse can reuse the links those can be scheduled within the interval of \( l_{x,y} \) or the link those satisfy Eq. (10). Therefore, the spatio-temporal bandwidth reuse can maximize the bandwidth reuse and thus completes the proof of lemma 1. □

The vertices, in Fig. 3(a), form a complete bi-partite graph [12]. The bi-partite graph has only one vertex (i.e., link, which is the mostly loaded link) in one partition, and \( n \) vertices in the second partition. We claim that the links corresponding to the vertices of a complete bi-partite graph can be scheduled simultaneously, and this scheduling achieves spatio-temporal bandwidth reuse.

**Lemma 2.** If \( n \) cliques (where, each clique is an induced subgraph in the BRG) of cardinality 2 have a common vertex and the remaining vertices form an independent set, then these \( n \) cliques always produce a complete bi-partite graph. Furthermore, the number of transmission units required to schedule all the links of a complete bi-partite graph is equal to the sum total of the loads of the mostly loaded partition.

**Proof.** In a complete bi-partite graph, a vertex in one partition is connected with all the vertices in another partition but is not connected with any vertex in its own partition. So, any link in one partition can be simultaneously scheduled with any link in the other partition and no two links in one partition can be simultaneously scheduled. The number of transmission units required to schedule the links of a partition is equal to the load of the partition. Therefore, once the links of the mostly loaded partition are scheduled, the links of the other partition can be scheduled by using spatio-temporal bandwidth reuse without any extra transmission units. This completes the proof of lemma 2. □

A generalization of scheduling the links as a complete bi-partite graph leads us to schedule the links as a complete \( r \)-partite graph [12]. Note that in a complete \( r \)-partite graph, two vertices from two different partitions are directly connected and can be scheduled simultaneously. We define the load of a partition as the sum of the loads of the links in that partition. Let \( d_r \) denote the loads of the mostly loaded partition. We need to allocate \( d_r \) transmission units to schedule all the links in the complete \( r \)-partite graph. If we schedule the mostly loaded partition, then links in the other partitions can be scheduled using spatio-temporal bandwidth reuse method.

**Lemma 3.** Consider a complete \( r \)-partite graph (which is an induced subgraph in the BRG), \( K_{m_1,m_2,...,m_r} \), where the number of vertices in the \( k \)-th partition is \( m_k \). Assume that the subgraph in the \( k \)-th partition is represented by \( P_k \) and the load of the \( k \)-th partition is \( L_k \). Then, spatio-temporal bandwidth reuse method can schedule the complete \( r \)-partite graph, where

- the number of transmission units required for the complete \( r \)-partite graph is \( d_r \).
- the gain\(^1\) achieved in the complete \( r \)-partite graph is \( \sum_{k=1}^{r} L_k - d_r \).

**Proof.** The load of the partition \( P_k \) is given by

\[
L_k = \sum_{l_{x,y} \in p_k} L_{x,y},
\]

where, \( p_k \) denotes the set of links (i.e., vertices) in the partition \( P_k \). Thus, the load of the mostly loaded partition is given by

\[
d_r = \max_{p_k \in \{m_1,m_2,...,m_r\}} (L_k).
\]

Once the links in the mostly loaded partition are scheduled, the links in the remaining partitions can be scheduled using spatio-temporal bandwidth reuse method. Since all the links in any other partitions are non-contending with the links of the mostly loaded partition, thus, the number of transmission units required for the whole complete \( r \)-partite graph is \( d_r \) and this completes the proof of the first part of lemma 3. Now, the total load of the complete \( r \)-partite graph is \( \sum_{k=1}^{r} L_k \). Since, we need \( d_r \) transmission units to schedule all the links of the complete \( r \)-partite graph, the gain, denoted as \( g(K_{m_1,m_2,...,m_r}) \), is \( \sum_{k=1}^{r} L_k - d_r \). This completes the proof of lemma 3. □

Note that if all the partitions of an \( r \)-partite graph have only one vertex, the \( r \)-partite graph becomes a clique of cardinality \( r \) and the gain will be equal to the gain calculated in [3], which exploits only spatial reuse. However, we argue that complete \( r \)-partite graphs with more than one vertex in one partition frequently exist in the BRG of mesh networks. For example, the BRG in Fig. 2 has two complete bi-partite graphs having more than one vertex in one partition (see Fig. 4(a) and Fig. 5(b)). Also, a mesh network of 10 SSs, SSs within the inner circle of the hexagonal topology used in simulation (see Fig. 9), contains a complete 3-partite graph having 2 vertices in two partitions. Therefore, the throughput can be further enhanced using spatio-temporal bandwidth reuse than that of only spatial bandwidth reuse.

5. **Fair Scheduling with Spatio-Temporal Bandwidth Reuse (FS\(^2\)BR)**

In this section, we explain the proposed scheduling algorithm, Fair Scheduling with Spatio-temporal Bandwidth Reuse (FS\(^2\)BR).
Reuse (FS²BR). It ensures the fairness explained in Sect. 4.1 and utilizes the bandwidth reuse explained in Sect. 4.2. To exploit the spatio-temporal bandwidth reuse, it searches all possible complete r-partite graphs in the BRG of a given mesh network. It selects the set of complete r-partite graphs, which produces the maximum gain, and consequently, maximizes the network throughput. The base algorithm finds the global optimal solution for the BRG, we call this Fair Scheduling with Spatio-temporal Bandwidth Reuse by Global-optimization (FS²BR-G). However, finding all possible combinations of complete r-partite graphs from a given BRG is an NP-class complexity problem [13]. Therefore, to reduce the complexity of such an exhaustive search of complete r-partite graphs, we propose two more algorithms, those partition the BRG into smaller subgraphs and then i) finds the optimal scheduling solution locally (FS²BR-LP), and ii) applies a greedy approach to find the optimal scheduling solution locally (FS²BR-GP). We present the aforementioned algorithms in the following subsections.

5.1 Fair Scheduling by Global Throughput Maximization (FS²BR-G)

The scheduling algorithm FS²BR-G, as its name suggests, searches the BRG to find all possible combinations of complete r-partite graphs. It selects a set of complete r-partite graphs as a feasible solution, if the set satisfies the following two conditions [3]

- The set of complete r-partite graphs covers all vertices in the BRG.
- A vertex does not appear in more than one complete r-partite graph, i.e., the set of complete r-partite graphs includes a vertex only once.

The first condition guarantees that a feasible scheduling set includes all the links in the BRM, i.e., if a link is active, the scheduling algorithm schedules it. Whereas, the second condition ensures that links are included only once, so that every link gets its exact share.

Suppose, FS²BR-G finds R feasible scheduling sets. Let $s_i$ denote the i-th set, where, $i = 1, 2, \ldots, R$. Note that each set $s_i$ has a gain, which is the sum of the gain of the complete r-partite graphs in the set. We denote the gain of the set $s_i$ by $g_i$ and the number of transmission units required for the set $s_i$ by $T_i$. Then, $g_i$ and $T_i$ are given by

$$g_i = \sum_{K_{m_1, m_2, \ldots, m_r} \in s_i} g(K^a_{m_1, m_2, \ldots, m_r})$$

$$T_i = \sum_{K_{m_1, m_2, \ldots, m_r} \in s_i} d^a_i$$

where $K^a_{m_1, m_2, \ldots, m_r}$ is the a-th complete r-partite graph and $d^a_i$ is its gain. We use the superscript $a$, since there can be many complete r-partite graphs having same number of partitions. The scheduler finds the set of complete r-partite graphs, which maximizes the gain or minimizes the number of transmission units in a round given in Eq. (11) or Eq. (12), respectively. Thus, the scheduling solution of FS²BR-G appears as an optimization problem and the optimization problem can be represented as

$$\text{maximize } g_i \text{ or minimize } T_i$$

subject to

$$\bigcup_{K_{m_1, m_2, \ldots, m_r} \in s} K_{m_1, m_2, \ldots, m_r} = U$$

$$K_{m_1, m_2, \ldots, m_r} \bigcap K^2_{m_1, m_2, \ldots, m_r} = \emptyset$$

$$\forall K^1_{m_1, m_2, \ldots, m_r}, K^2_{m_1, m_2, \ldots, m_r} \in s$$

The detailed scheduling algorithm is summarized in Algorithm 1. Step 1 finds all possible complete r-partite graphs from the BRG. Step 2 solves the optimization problem given in Eq. (13).

Algorithm 1 FS²BR-G

Input : Bandwidth reuse matrix, $M[n][n]$
Load Table, Load$r$

Step 1: Search all complete r-partite graphs in the BRG.
Step 2: Solve the optimization problem given in Eq. (13).

5.2 Fair Scheduling by Graph Partitioning and Local Throughput Maximization (FS²BR-LP)

The key idea of FS²BR-LP is to partition the BRG into smaller subgraphs and to apply the FS²BR algorithm on each subgraph to find the optimal scheduling, i.e., to find a local optimal solution instead of a global optimal solution. However, the partitioning algorithm does not partition the BRG at a time, rather it uses progressive partitioning. The partitioning algorithm finds an induced subgraph in $G$, using the following steps:

- First, it selects the vertex in $G$, $v$, corresponding to the mostly loaded link in the BRM.
- Second, it finds the set of neighbors of $v$, denoted as $V$.
- Finally, it finds the induced subgraph, $G'$, made by $v$ and its neighbors, $V$. The induced subgraph is the partition, where FS²BR algorithm is applied to find the local optimal solution.

The scheduler allocates a block of $L_v$ transmission units for the induced subgraph, where $L_v$ is the load of the vertex $v$. Then, the scheduler applies the FS²BR algorithm to find the local optimum scheduling. If all the links form a single complete r-partite graph, and the load of the mostly loaded partition is $L_v$, then the whole subgraph can be scheduled within $L_v$ transmission units. Otherwise, a single complete r-partite graph is scheduled, which has the maximum gain.
Algorithm 2 FS2BR-LP

Input: Bandwidth Reuse matrix, $M[n][n]$  
Load Table, $Load[n]$  
Transmission unit, $tu$, $tu_1$

Initialization: $tu = 1$

Step 1: Select the vertex $v_i$ corresponding to the link which has the highest load in the BRM.

Step 2: Find the set of vertices, $V = [v_1, v_2, \ldots, v_n]$ in $G$, which are neighbors of $v_i$.

Step 3: Find the induced subgraph $G'$ in $G$ containing the vertex $v_i$ and the set of vertices $V$.

Step 4: Find the $r$-partite graph, $K_{s_1,s_2,\ldots,s_r}$ in $G'$ having the highest gain.

Allocate $tu$ to $tu + Load[v_i]$ to link $v_i$.

for each vertex $v_i$ in $V$ do

if $tu_1 + Load[v_i] \leq tu + Load[v_i]$ then

Allocate $tu_1$ to $tu_1 + Load[v_i]$ transmission slot to $v_i$

$tu_1 = tu_1 + Load[v_i]$

$Load[v_i] = 0$

else

Allocate $tu_1$ to $tu_1 + Load[v_i]$ to $v_i$

$Load[v_i] = Load[v_i] - (tu + Load[v_i] - tu_1)$

BREAK

end if

end for

for each vertex $v_i$ in $V$ do

if $Load[v_i] = 0$ then

$G = G - v_i$

end if

end for

end for

Step 6: if number of vertices in $G = 0$ then

STOP

else

GOTO Step 1.

end if

If the load of the mostly loaded partition of the complete $r$-partite graph is greater than $L$, then some links of the complete $r$-partite graph are partially scheduled. The scheduler then deletes the already scheduled vertices (i.e., the vertices which have zero load after scheduling) of $G$ one after another. Therefore, after scheduling the vertices of one subgraph (i.e., allocating transmission units for the vertices and deleting them from $G$), $G$ becomes a vertex deleted graph [12], that is, $G = G - v_i$, where, $v_i$ is a vertex in $G'$ and after scheduling the vertices of $G'$, we have $L_0 = 0$.

Such a partitioning algorithm that uses local optimization has two benefits: i) since it searches the optimal solution within a smaller graph, or more specifically, within a graph where one vertex is the neighbor of all the other vertices, it reduces the complexity of searching the complete $r$-partite graphs, and ii) when the vertices of the induced subgraph (the partition which has just scheduled) are deleted, it decreases the number of edges in the remaining graph, since it deletes not only the edges in $G'$ but also the edges that have an end-point in $G'$. However, it is true that if we delete a vertex from $G$ after allocating its fair share of

5.2.1 An Example of Scheduling by FS2BR-LP

In this subsection, we explain how FS2BR-LP allocates the transmission units fairly and maximizes the network throughput. This example shows the scheduling of the BRG in Fig. 2, where the loads of the vertices (1, 0), (6, 0), (7, 6), (2, 1), (4, 1), (3, 2), (5, 4), (8, 7) and (9, 7) are 7, 6, 5, 3, 3, 2, 2, 2 and 2, respectively. The load of the vertices are sorted in non-increasing order. In that graph, the vertex (1, 0) is the mostly loaded vertex and its one-hop neighbors are (8, 7) and (9, 7). The scheduling algorithm takes the induced subgraph containing these vertices and applies the FS2BR algorithm to find the optimum scheduling. FS2BR finds the complete bi-partite graph shown in Fig. 4(a). One partition of the complete bi-partite graph has only one ver-
tex (link (1, 0)) and is the mostly loaded link in the complete bi-partite graph. Since \( L_{1,0} > L_{8,7} + L_{9,7} \), the whole subgraph requires \( L_{1,0} \) transmission units. However, if we would schedule using clique construction, then it would require \( L_{1,0} + L_{8,7} \) transmission units where, \( L_{8,7} \geq L_{9,7} \).

Figure 4(b) shows graph \( G \), after deleting the vertices \((1, 0), (8, 7)\) and \((9, 7)\) and the links which have at least one end point to these vertices. The partitioning algorithm then finds the induced subgraph shown in Fig. 5(a). Since the subgraph constitutes a clique of cardinality 3, the scheduler schedules the cliques.

The scheduler then starts with the vertex \((7, 6)\) and includes the remaining two vertices in the induced subgraph (Fig. 5(b)). Once again \( \text{FS}^2\text{BR} \) finds a complete bi-partite graph but \( L_{7,6} < L_{2,1} + L_{4,1} \). So, it cannot schedule both \((2,1)\) and \((4,1)\). However, \( \text{FS}^2\text{BR} \) schedules link \((2,1)\) and allocates 2 transmission units to \((4,1)\). Therefore, link \((4,1)\) remains with a load of ‘1’ transmission unit. Since there is no other one-hop neighbor of \((4,1)\), the scheduler then allocates 1 transmission unit for \((4,1)\). Figure 6 shows the allocated transmission units to the links.

5.3 Fair Scheduling by Graph Partitioning and Greedy Throughput Maximization (\( \text{FS}^2\text{BR-GP} \))

\( \text{FS}^2\text{BR-GP} \) partitions the BRG into smaller subgraphs in a similar way as explained in Sect. 5.2. However, it finds the local optimal solution for the subgraphs using a greedy approach. The detailed algorithm is presented in Algorithm 3. We assume that the links in the BRG are sorted and the load table stores the load of the links in the same order as it is in the BRG.

Line 7 selects the mostly loaded link in the graph. Lines 8–10 check the load of the selected link and the algorithm skips the link, if its load is zero. Lines 12–14 allocate transmission units for this link, which is equal to the load \( L_{v} \) of the link. Also, the number of transmission units allocated for this subgraph is \( L_{u} \). Line 17 picks every link one after another and line 18 selects whether the link is a neighbor of the mostly loaded link. Instead of finding the optimal solution for the subgraph, it approaches to find the optimal solution for each neighbor. Since the neighbors are also stored in the decreasing order of their loads, it first finds the neighbor with the highest load. Note that each of the neighbors can be individually scheduled in parallel with the mostly loaded link of the subgraph. Lines 19–23 allocate transmission units for the first neighbor. The algorithm assigns trans-

**Algorithm 3 FS\(^2\)BR-GP**

1: Input : Bandwidth Reuse matrix, \( M[n][n] \)
2: Load Table, \( \text{Load}[n] \)
3: Variable: \( n \) - number of links
4: \( \text{count} \) - number of transmission units for a round
5: \( \text{block} \) - number of transmission units for a subgraph
6: ASSIGN a value, –1, to each entry of \( \text{SCH}[n][\text{count}] \)
7: \( (\text{–1}) \) will be used as a termination condition for a slot
8: Initialize: \( \text{count} = 1 \)
9: for EACH link \( i = 1 \) TO \( n \) do [allocate slot for each link]
10: if \( \text{LOAD}[i] == 0 \) then (this link is already scheduled)
11: CONTINUE with the next link
12: end if
13: (allocate transmission unit to the mostly loaded link)
14: for EACH slot \( j = \text{count TO count + LOAD}[i] \) do
15: \( \text{SCH}[1][j] = i \)
16: end for
17: \( \text{flag} = 1 \) [track mostly loaded (first) neighbor of \( i \)]
18: (schedule first neighbor of \( i \) without checking)
19: for EACH link \( j = 1 \) TO \( n \) do
20: if \( \text{M}[i][j] == 1 \) then
21: if \( \text{flag} == 1 \) then
22: for \( k = \text{count TO count + LOAD}[j] \) do
23: \( \text{SCH}[2][k] = j \)
24: end for
25: end for
26: \( \text{LOAD}[j] = \text{flag} = 0 \)
27: else
28: block = count
29: while \( \text{block < count + LOAD}[i] \) do
30: \( \text{row} = 2 \)
31: \( \text{flag} = 1 \)
32: (Check \( j \) contends with allocated links)
33: while \( \text{SCH}[\text{row}][\text{block}] != -1 \) do
34: if \( \text{M}[\text{SCH}[\text{row}][\text{block}]][j] == 1 \) then
35: \( \text{row} += + \)
36: block = 0
37: end if
38: end while
39: \( \text{LOAD}[j] = \text{LOAD}[j] - 1 \)
40: end if
41: \( \text{flag} = 0 \)
42: BREAK
43: end if
44: \( \text{block} += + \)
45: end while
46: end if
47: end if
48: end if
49: end for
50: \( \text{count} = \text{count} + \text{Load}[i] \)
51: \( \text{LOAD}[i] = 0 \) (since link \( i \) is already scheduled)
52: (delete the completely scheduled vertex and its links)
53: for each slot \( k = 1 \) TO \( n \) do
54: if \( \text{M}[i][k] == 1 \) AND \( \text{LOAD}[k] == 0 \) then
55: for \( m = 1 \) TO \( n \) do
56: \( \text{M}[m][k] = \text{M}[k][m] = 0 \)
57: end for
58: end if
59: end for
60: end for

Fig. 6 Throughput maximized TDMA scheduling for the example network shown in Fig. 1 by the proposed \( \text{FS}^2\text{BR-LP} \) algorithm, which uses spatio-temporal bandwidth reuse.
mission units to the first neighbor without checking whether this link contends with the already allocated links, since the only allocated link is the mostly loaded link and its neighbor in the BRG.

Lines 25–46 allocate transmission units to other neighbors (except the first neighbor) in a greedy approach. It starts with the first transmission unit in the block of transmission units allocated for this subgraph, and checks the links so far allocated for this transmission unit in lines 30–37. If all the allocated links are non-contending with the current link, then this slot is allocated for the current link. Lines 38–41 allocate a transmission unit to the current link and decrement the load of the link by ‘1.’ Otherwise, the algorithm moves to the next transmission unit and continues until the required number of transmission units is allocated for the current link (lines 42–44) or it finishes the entire block of transmission units allocated for this subgraph.

The algorithm continues the above greedy approach for all the links in subgraph. In line 50, the value of count is incremented by \( L_{\text{max}} \). In line 51, the load of the mostly loaded link is assigned to zero. Lines 53–59 delete the links of the subgraph one after another, from the original graph, if the load of the link is zero; that is, the link is scheduled completely.

One important property of the greedy approach is that it can apply the spatio-temporal bandwidth reuse for a special type of \( r \)-partite graph, which is not complete. For example, consider the \( r \)-partite graph shown in Fig. 7(a). Suppose, links \( a, b, c, d \) and \( e \) have loads 10, 6, 5, 4 and 4, respectively. There is no complete \( r \)-partite graph in the subgraph but there are two cliques of cardinality 3. So, the algorithms in Sects. 5.1 and 5.2 first schedule the clique having links \( a, b, \) and \( c \), and then, schedule the clique having links \( d \) and \( e \). However, the algorithm in Sect. 5.3 schedules the links exploiting spatio-temporal bandwidth reuse.

Links \( a, b \) and \( c \) form a clique and exploit the spatial bandwidth reuse. All three proposed algorithms schedule them, but the greedy approach finds that the transmission units 7–10 are only allocated for link \( a \) and both link \( d \) and \( e \) can be scheduled with link \( a \) as they form another clique. Also, the loads of both \( d \) and \( e \) are equal to the transmission units in 7–10. So, it schedules \( d \) and \( e \) using spatio-temporal bandwidth reuse. Note that if the load of \( d \) and \( e \) are more than 4, then, it allocates 4 transmission units to both \( d \) and \( e \) within the transmission units 7–10.

6. QoS-Aware Fair Scheduling for IEEE 802.16 Mesh Networks

In Sect. 5, we have proposed the fair scheduling and throughput maximization algorithms for wireless mesh networks, where the flows/connections have no specific demand. Therefore, the flows have been allocated equal transmission opportunity in a transmission round. In this section, we extend the proposed algorithms, so that, they can be applied for IEEE 802.16 based mesh networks with different traffic classes. Note that IEEE 802.16 allocates transmission opportunity in slots as the smallest unit; therefore, from now on we will consider a transmission round in terms of number of slots instead of number of transmission units. Also, in IEEE 802.16, time is partitioned into frames of fixed size, each having fixed number of slots for data transmission (for a detailed description of the frame structure, see [11]). The duration of a transmission round (i.e., the number of slots in a transmission round) is an implementation dependent issue. For example, a frame can be considered as one transmission round and slots of the frame are fairly distributed among the active flows or a fixed number of slots can be allocated for a flow, and thus, a transmission round can use one or more frames depending on the size of the allocation and the number of active flows. Since the proposed algorithm can work independently, irrespective of the duration of a transmission round, we will not address this issue further.

As explained in Sect. 3.1, four traffic classes are defined for IEEE 802.16, and they have different demands from the networks. But, based on the demands, only the number of slots allocated for a connection will vary. Thus, the loads of the links that carry the traffic of the connections will be different. However, the BRM usually does not change with the slot allocation unless otherwise, the allocation makes the load of one or more links to zero. Therefore, once the scheduler calculates the loads of the active links, the algorithms presented in Sect. 5 are directly applicable. Thus, the scheduling algorithm needs to calculate the load of the links based on the specific demands and the routes (i.e., path in the mesh network) of the connections. Therefore, QoS-awareness simply tells how the demands of a connection can be converted to the number of slots allocated for that connection for one or more links in a frame. A detailed slot allocation mechanism is presented in [15] for different traffic classes of IEEE 802.16 in PMP mode.

In the following subsections, we explain how the slots of a frame can be allocated to the connections of different traffic classes, if connections of one or more traffic classes are present in the network.

6.1 Slot Allocation for Best-Effort (BE) Traffic

Best-Effort traffic does not have any specific demand and so the algorithms, presented in Sect. 5, are directly applicable, if traffic of other classes is not present in the network. The minimum allocation for BE traffic can be zero and the max-
imum allocation should not exceed the amount of request. To send bandwidth request, traffic of BE class participates in the contention.

6.2 Slot Allocation for nrtPS Traffic

As mentioned in the standard, nrtPS traffic class demands a better than best effort service, which can be supported either by assigning weight for those connections or allocating a minimum number of slots (after fulfilling the demand of the higher classes of traffic, if there is any) in each round for them, depending on the implementation. If both BE and nrtPS traffics are present, the assignment of weight will provide better service to nrtPS traffics than the BE traffics. A different implement could assign minimum number of required slots for each nrtPS connection and then equally distribute the remaining slots to both the nrtPS and BE traffics, with the exception that the allocation should not exceed the requirement of the traffic of either class. Finally, the scheduler calculates the load of each link based on the assignment. To send bandwidth request, the BS allocates slot(s) to the traffic of nrtPS class (for multihop flows, one slot in each link) in one transmission round within a given number of transmission rounds (nrtPS Poll interval).

6.3 Slot Allocation for rtPS Traffic

The rtPS traffic class has specific demand from the networks and they cannot participate in the contention. The BS should allocate slot(s) for the flows of this traffic class in each round, so that they can send at least one slot of data (i.e., bandwidth request) to the BS. Thus, if a flow is h hops away, the BS must have to allocate one slot to each of the h links for that flow. The rtPS connections are guaranteed service, so the scheduler should first allocate the required slots for them and after that fairly allocate the remaining slots to the nrtPS and BE traffic. More specifically, a scheduler can divide the transmission round in two parts, one for the guaranteed service and other for the nrtPS and BE traffics. In the worst case, guaranteed service part could cover the whole part of a particular transmission round.

6.4 Slot Allocation for UGS Traffic

Unlike other service classes, the UGS class does not send any bandwidth requests and cannot participate in the contention and they require a fixed number of slots. Therefore, the scheduler allocates fixed number of slots for them.

7. Performance Analysis and Evaluation

7.1 Analytical Analysis

7.1.1 Fairness and Throughput Analysis

Proposition 1. The proposed scheduling algorithms are fair, that is, they allocate fair bandwidth to all active flows.

Proof. The conditions mentioned in Sect. 5.1 and the optimization problem in Eq. (13) ensure that during a transmission round all active links are scheduled and every single link is scheduled only once. Since the load of a link calculated in Eq. (4) ensures that a link is allocated one transmission unit in a round for every flow forwarded by the link, so every flow gets one transmission unit in a round in all links of the path of that flow. Therefore, the BS receives exactly one transmission unit from each flow in a round. Thus, the proposed algorithm in Sect. 5.1 is fair. Also, the algorithms in Sects. 5.2 and 5.3, allocate transmission units in a way so that every link gets transmission opportunity in a round which is proportional to their loads. So, both these two algorithms are fair as well.

Proposition 2. The proposed scheduling algorithms are collision-free, that is, they allocate collision-free transmission unit(s) to each link.

Proof. The algorithms in Sects. 5.1 and 5.2 schedule the links as a disjoint unit of complete r-partite graphs. Note that all links in a complete r-partite graph can be scheduled without collision, if the allocated transmission units are equal to the load of the mostly loaded partition of the complete r-partite graph (see lemma 3 for details). Also, two links in two different complete r-partite graphs are scheduled in two different time interval. Therefore, FS^2BR-G and FS^2BR-LP are collision-free scheduling.

The scheduling in FS^2BR-GP, before allocating a transmission unit to a link, checks whether all the links allocated so far for that transmission unit is non-contending with the current link (the link which is currently under scheduling) or not. If all the links allocated for a transmission unit are non-contending, only then, FS^2BR-GP allocates the transmission unit to the current link. Thus, the scheduling in FS^2BR-GP is also collision-free.

Proposition 3. The progressive partitioning of a graph into a smaller subgraph (i.e., the induced subgraph with the mostly loaded vertex of the graph and its neighbors) and local scheduling of the links of the subgraph using spatio-temporal bandwidth reuse, by the algorithms FS^2BR-LP and FS^2BR-GP, achieve as optimal bandwidth reuse as the global optimum bandwidth reuse of FS^2BR-G.

Proof. The proposed graph partitioning algorithm starts with the mostly loaded vertex and includes all the neighbors of the mostly loaded vertex. Then, a block of transmission units is allocated for the subgraph, which is equal to the load of the mostly loaded vertex. Since spatio-temporal bandwidth reuse (or only spatial reuse of bandwidth) can be applied to the links, which are non-contending and are neighbor in the BRG, therefore, these links are already included in the subgraph; so, the graph partitioning achieves the local optimal bandwidth reuse. On the other hand, it is not possible to schedule a link with less number of transmission units than its load, if there is no non-contending link having higher load. If there would have been any of such links only
then, the link could utilize the bandwidth of that link. Therefore, a scheduling algorithm has to allocate at least the load of the mostly loaded link as the block of transmission units for that subgraph. So, the scheduling algorithms (FS\textsuperscript{2}BR-LP and FS\textsuperscript{2}BR-GP) achieve the same bandwidth reuse as the global optimum bandwidth reuse of FS\textsuperscript{2}BR-G. Simulation results presented in Sect. 7.2.1 also justify that. However, FS\textsuperscript{2}BR-GP might outperform the other two algorithms in some specific topologies, as explained in Sect. 5.3.

7.1.2 Complexity Analysis

All three algorithms proposed in this paper require the BRM as an input, which has polynomial complexity. FS\textsuperscript{2}BR-G algorithm needs to find all possible combination of complete r-partite graph in the BRG, which is an NP-class problem. FS\textsuperscript{2}BR-G algorithm also needs to solve the optimization problem in Eq. (13), even though heuristic approach can be used to solve this. Therefore, we proposed two more algorithms.

FS\textsuperscript{2}BR-LP searches the complete r-partite graphs within a small subgraph, which reduces the complexity of searching. However, the main benefit is that the smaller graph starts with the mostly loaded link of the graph and the remaining links of the smaller graph are non-contending with the mostly loaded link. Also, the subgraph cannot allocate transmission units more than the load of the mostly loaded link. So intuitively, it can be said that there will be only one complete r-partite graph, that produces the maximum gain and includes the mostly loaded link. This simplifies the searching problem, because now we need to search a single complete r-partite graph.

FS\textsuperscript{2}BR-GP algorithm explained in Sect. 5.3 does not schedule the links based on complete r-partite graph (or clique), rather it allocates the transmission units in a greedy approach within a smaller partition and has a polynomial complexity. The scheduling principle is based on complete r-partite graph and so utilizes spatio-temporal bandwidth reuse.

7.2 Simulation Results

We have evaluated the efficiency of the proposed scheduling algorithms and the QoS achieved by the flows of different traffic classes. We have developed a program in C++ to evaluate the scheduling efficiency of the proposed algorithms and the QoS achieved by the flows. The parameters and MAC implementation contain the main features of IEEE 802.16 standard. All the SSs use same MCS (16-QAM 3/4).

Table 1 shows the simulation parameters in detail. We consider that the channel is error-free. In the simulation, we assume that two links are interfering, if i) either one of the receiving SSs of the two links is the neighbor of both the sending SSs, ii) either the sending SSs or the receiving SSs are neighbor of each other. Also, we assume that an SS can either send or receive in a given slot and it can send to or receive from a single SS in the slot.

7.2.1 Scheduling Efficiency

The efficiency of the scheduling algorithms depends on the topology of the mesh network and number of connections. Therefore, we conducted the simulation on two different topologies of mesh networks, where i) the mesh network forms a chain topology, that is, a one dimensional mesh network (Fig. 8), and ii) the mesh network forms a hexagonal topology, that is, a two dimensional mesh network (Fig. 9).

We measure the number of transmission units required for a single transmission round for each simulation set up in a specific topology with different number of SSs and connections. Then, we compare the results of the three proposed algorithms with standard TDMA (no bandwidth reuse) and the clique construction algorithm in [3], S-TDMA (which uses spatial reuse of bandwidth).

Figure 10 shows the simulation results for the chain topology for different number of SSs. Number of transmission units required for a single transmission round for each simulation set up in a specific topology with different number of SSs and connections.
results. In a chain topology, two links can be simultaneously scheduled, if there are two or more links in between them. Also, all the traffic go through the same path and loads of the links gradually decrease with the increase of the distance from the base station. Therefore, in chain topology, three separate cliques can be constructed and these three cliques include all the links of the chain. Figure 11 shows that the clique construction algorithm in [3], constructs three cliques and schedules all the links. Note that the first, second and third cliques include links \( l_{1,0}, l_{2,1} \) and \( l_{3,2} \), respectively. Since, any two of these links interfere with each other, this is the optimum scheduling.

Figure 12 shows that the proposed greedy algorithm also achieves the optimum scheduling. But, it uses the spatio-temporal bandwidth reuse and schedules all the links within 38 transmission units except link \( l_{3,2} \). That is, not only it achieves the optimum scheduling but also it can schedule more links in parallel; though, due to the topology, it requires same number of transmission units as in Fig. 11.

Figure 13 shows the simulation results for the hexagonal topology (Fig. 9) and all three proposed algorithms outperform the traditional TDMA and the S-TDMA mechanism. The spatio-temporal bandwidth reuse mechanism requires lower number of transmission units in a round. This justifies that if the links are scheduled as a complete r-partite graph instead of a clique, the bandwidth utilization increases. Also, note that the greedy algorithm can schedule some r-partite graphs, which are not complete; and thus, achieves better result than the other two proposed algorithms. Figure 13 also shows that FS\(^2\)BR-G and FS\(^2\)BR-LP produce the same result, that is, the local throughput optimization by graph partitioning produces the global optimum value.

Figure 14 shows scheduling of 10 SSs in the inner circle of the hexagonal topology (Fig. 9) by using the clique construction algorithm. The scheduling algorithm selects four cliques with the maximum gain, where one clique has just one link. Figure 15 shows the same scheduling solution of FS\(^2\)BR-G and FS\(^2\)BR-LP algorithms. Both the two algorithms schedule links \((3, 0), (4, 1), (5, 1), (6, 2)\) and \((7, 2)\) within 8 transmission units. Note that links \((4, 1)\) and \((5, 1)\) can be scheduled separately with both \((6, 2)\) and \((7, 2)\), but they cannot be scheduled simultaneously. Similarly, links \((6, 2)\) and \((7, 2)\) can be scheduled separately with both \((4, 1)\) and \((5, 1)\) but they cannot be scheduled simultaneously. Thus, these 5 links form a complete r-partite graph and so, both the proposed algorithms use the spatio-temporal bandwidth reuse. The remaining links form two complete r-partite graphs (i.e., two cliques) and are sched-
Figure 16 shows the scheduling solution for the 10 SSs of the inner circle of the hexagonal topology by FS^3BR-GP algorithm. It first schedules the complete r-partite graph similar to the other two algorithms. But, it also schedules an r-partite graph which is not complete and achieves spatio-temporal bandwidth reuse. This r-partite graph has 3 partitions. One partition contains the mostly loaded link of the r-partite graph, link (1, 0), another partition contains link (8, 3) and the last partition contains links (9, 3) and (10, 9). It first schedules link (1, 0) and then link (9, 3). A greedy search allows the allocation of the links (8, 3) and (10, 9). Thus, the greedy algorithm, in this way, can schedule an r-partite graph, which is not complete.

7.2.2 QoS-Aware Fair Scheduling

We use the network topology shown in Fig. 1 to evaluate the QoS achieved by the flows. We assume that each SS has four flows/connections of four different traffic classes, explained in Sect. 6. QoS requirements of the flows of different QoS types are given in Table 2. We assume that all the flows of a particular traffic class have same requirements. All the SSs are assumed to be entered the network in the beginning and during the simulations all the flows are connected.

<table>
<thead>
<tr>
<th>Traffic Class</th>
<th>Packet Size (bytes)</th>
<th>Bandwidth (bps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>UGS</td>
<td>206</td>
<td>min 82,400 max</td>
</tr>
<tr>
<td>rtPS</td>
<td>500</td>
<td>480,000 640,000</td>
</tr>
<tr>
<td>nrtPS</td>
<td>500</td>
<td>120,000 160,000</td>
</tr>
<tr>
<td>BE</td>
<td>200</td>
<td>- 120,000</td>
</tr>
</tbody>
</table>

Table 2 QoS requirements of different traffic classes.

Figure 17 shows the throughput achieved by the flows of different traffic classes. In the simulation, we assume that UGS and rtPS flows in odd and even number of SSs start to send data at time 0 and 20-th second, respectively. Whereas, nrtPS and BE flows in odd and even number of SSs start to send data at time 10 and 30-th second, respectively. Therefore, for the interval 30–39 seconds, all the flows from all the SSs are sending data. Figure 17 shows that the BS allocates a sufficient number of slots to the UGS flows and the throughput of these flows is always 0.0824 Mbps. The throughput of the rtPS flows depends on the variable-rate source data. The BS always allocates the required bandwidth to the rtPS flows. The throughput of the nrtPS and BE flows is bounded by the maximum requirements. The throughput of the BE flows depends on the free resources left after ensuring the QoS requirements of all the classes. Thus, the throughput of the BE flows depends on the network load as justified in Fig. 17.

Figure 18 shows the throughput of the individual flows, during the interval of 30–39 seconds, where all the flows from all the SSs are active. As the figure shows, the proposed scheduling mechanism fairly allocates the bandwidth to the active flows.

The fairness and delay of the flows under different loads are measured using the same simulation set up used for Fig. 17. Since the throughput and delay of both the UGS and rtPS flows are fulfilled irrespective of the network loads, we only show the throughput and delay of nrtPS and BE traffic. Figure 19 and Fig. 20 show results for the time interval 10–19 seconds (lightly loaded) and 30–39 seconds (loaded), respectively. The figures indicate that an increase in the network load decreases the throughput and increases the delay.
of the flows. But, the throughput achieved by the flows of same class is equal. Thus, the proposed mechanism provides fairness, even with varying loads.

Figure 21 shows the scheduling delay of different traffic classes with varying hop count for the network in Fig. 9. The scheduling delay is measured for two different cases. In the first case, duration of a transmission round is 1 frame, and in the second case, duration of a transmission round is 3 frames. The scheduling delay is zero for the UGS flows. For other classes delay might vary with network load. The rtPS flows are allocated at least one slot in each link, so the scheduling delay is at most one round and is constant. The nrtPS flows can send request in one round in a given number of rounds (nrtPS Poll interval). We use 5 as nrtPS Poll interval. Thus, the expected delay of nrtPS flows are around 2.5 rounds. Finally, BE traffic uses contention based bandwidth request and the scheduling delay varies with hop count of the flows.

The maximum achievable hop counts for a QoS-aware flow depends on the length of the transmission round (mostly determined by the load of the network, topology of the network, packet size of the flows and scheduling efficiency) and the allocation of the transmission units to the links. However, the allocation of transmission units to the links determines the delay of a packet in a node, and thus, the end-to-end delay of QoS-aware flows might not be satisfied beyond a certain hop count. If the allocated forwarding transmission units of a node appear before the reception units in a transmission round, in the worst case, the delay of a packet at any node might be 1 transmission round. In this case, the maximum achievable hop count is bounded by the number of transmission rounds within the delay deadline. On the other hand, a careful allocation and ordering of transmission units might allow a node to receive and forward the packet in the same round. If this is true for all forwarding nodes of a flow, the maximum end-to-end delay of a packet is 1 transmission round. Moreover, a discrete allocation of transmission units to the links (alternate reception and forwarding of packets by the nodes as mentioned in [2]) might allow a node to immediately forward a packet just after reception. In this paper, our intention is to maximize the network throughput and assign the required bandwidth to the QoS-aware flows. Thus, we only ensure that an upstream node gets the allocation of transmission units before a downstream node. Though, a perfect reordering of the transmission units, where each node alternately receive and forward a packet, might ensure minimum delay at any node. However, such an allocation might need a sophisticated algorithm which minimizes the delay using optimization [2], which we leaves as a future work.

The scheduling cost of a connection depends on whether the connection sends the bandwidth request implicitly or explicitly to send newly arrived data. In implicit bandwidth request, connections piggyback their bandwidth request with data, and hence, the scheduling cost is negligible. Whereas, when the queue of a connection is empty, connections need to send the bandwidth request explicitly. The scheduling cost for UGS connection is zero. To guarantee the QoS, the rtPS connection requires dedicated slot(s), so, the cost is proportional to the hop count. The nrtPS connection requires the same cost as the rtPS connection, whereas BE traffic requires little higher cost due to the contention. However, in a loaded network, both BE and nrtPS traffic are expected to be backlogged almost all the time and thus do not need to send explicit request. On the other hand, in a lightly loaded network, connections might need explicit request and due to the light load, the network can afford this.

8. Conclusion

In this paper, we have designed a centralized fair scheduling scheme, which ensures per connection fairness and enhances the throughput for IEEE 802.16 mesh networks. We investigated the state-of-the-art techniques (e.g., spatial reuse) for throughput enhancement and identified that the network throughput depends on the efficient scheduling of the links both in spatial and time domain. Then, we proposed the spatio-temporal bandwidth reuse method that achieves a higher network throughput and provides fairness. Also, we proposed a greedy scheduling algorithm that runs in polynomial time complexity. We extended the proposed scheduling mechanism to support the IEEE 802.16 QoS traffic classes. Simulation results show that our scheme outperforms existing methods in terms of throughput and guarantees the QoS of the connections.

References


Md. Abdul Hamid received his B.Engg. in Computer & Information Engineering in 2001 from International Islamic University Malaysia (IIUM). He received his Ph.D. from Kyung Hee University, South Korea in 2009. In 2002, he joined as a lecturer in the Computer Science & Engineering Department, Asian University of Bangladesh, Dhaka. Currently, he is working as a full-time lecturer in the Department of Information and Communications Engineering at HANKUK University of Foreign Studies, South Korea. His research interest includes Security in Wireless Mobile Ad Hoc and Sensor Networks, especially on Secure routing and Key management in Wireless Sensor Networks.

Md. Abdul Razzaque received the B.S. degree in Applied Physics and Electronics and M.S. degree in Computer Science from the University of Dhaka, Bangladesh in 1997 and 1999 respectively. He received his Ph.D. from Kyung Hee University, South Korea in 2009. His research interest is in the area of modeling, analysis and optimization of protocols and architectures for wireless sensor networks, distributed systems, etc. Currently, he is working as a post doctoral researcher at Kyung Hee University.

Choong Seon Hong received his B.S. and M.S. degrees in electronic engineering from Kyung Hee University, Seoul, Korea, in 1983, 1985, respectively. In 1988 he joined KT, where he worked on Broadband Networks as a member of the technical staff. From September 1993, he joined Keio University, Japan. He received the Ph.D. degree at Keio University in March 1997. He had worked for the Telecommunications Network Lab, KT as a senior member of technical staff and as a director of the networking research team until August 1999. Since September 1999, he has been working as a professor of the School of Electronics and Information, Kyung Hee University. He has served as a Program Committee Member and an Organizing Committee Member for International conferences such as NOMS, IM, APNOMS, E2EMON, CCNC, ADSN, ICPP, DIM, WISA, BeN and TINA. His research interests include ad hoc networks, network security and network management. He is a member of IEEE, IPSJ, KIPS, KICS and KIHEE.

Muhammad Mahbub Alam received his B.S. degree in Applied Physics and Electronics and M.S. degree in Computer Science from the University of Dhaka, Bangladesh in 1998 and 2000, respectively. He received his Ph.D. from Kyung Hee University, South Korea in 2009. Since 2002, he has been working as a faculty member in the Computer Science and Information Technology (CIT) department in Islamic University of Technology (IUT), Dhaka, Bangladesh. Currently, he is working as a post doctoral researcher at Kyung Hee University, South Korea. His research interests include wireless and mobile networking and performance modeling and analysis of networking systems.