Chapter 2

Stability-Based Topology Control in Wireless Mesh Networks

Gustavo Vejarano

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/48576

1. Introduction

Topology control[1] in wireless mesh networks is an important problem due to the effects it has on the different layers of the protocol stack [1]. For example, the network connectivity, energy consumption, total physical-link throughput, spatial reuse, and total end-to-end throughput as a function of the network topology have been investigated in [2-6] respectively. In this chapter, we look at the problem of topology control for adapting the stability region of the link-scheduling policy of the network. Therefore, we start by defining the problem of link scheduling and the stability region.

The goal when designing link-scheduling policies is to achieve maximum throughput while making the policies amenable for implementation [7, 8]. Link scheduling refers to the selection of a subset of links for simultaneous transmission that have the following characteristic: When the links are activated simultaneously, the interference between them is low enough to allow successful reception for every activated link. A link-scheduling policy specifies the mechanism that determines, for every time slot, a subset of links that fits this characteristic. For example, consider the network and the link (i, j) shown in Figure 1. Let this network operate under the frame structure shown in Figure 2. Therefore, in the network, time is divided into frames; each frame is divided into a control subframe and a data subframe, and each subframe is further divided into a series of time slots. Whenever link (i, j) is activated by the link-scheduling policy during a data-time slot, the link transmits a data packet. In order for the packet to be received successfully, none of the links that interfere with (i, j) can be active while (i, j) is active. Otherwise, the packet transmitted by node i is not received successfully by node j. This is known as a packet collision at

[1]In this chapter, topology control refers to the problem of controlling the creation and elimination of wireless links and the interference between them by controlling the transmission power of the nodes.

node j, i.e., the packet transmitted over (i,j) collides with the packet transmitted over the interfering link. The set of links that interfere with (i,j) is denoted by $\mathcal{I}^{(i,j)}$ in Figure 1. Therefore, when (i,j) is active, none of the links in $\mathcal{I}^{(i,j)}$ can be active. Given that every link has a set of interfering links, only subsets of the set of all links in the network can be active at a given time. The task of the link scheduling policy is to select one of these subsets for every data-time slot. This selection is done by exchanging control information during the control-time slots.

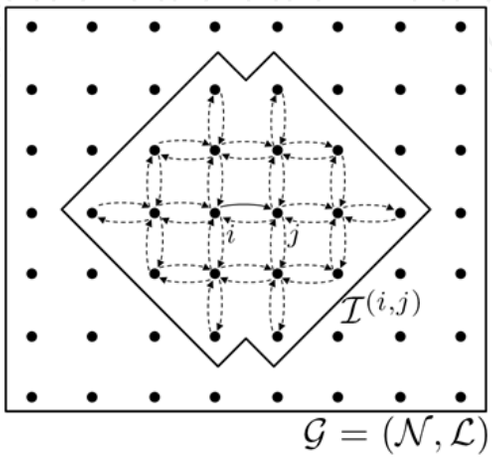

Figure 1. Interfering Links

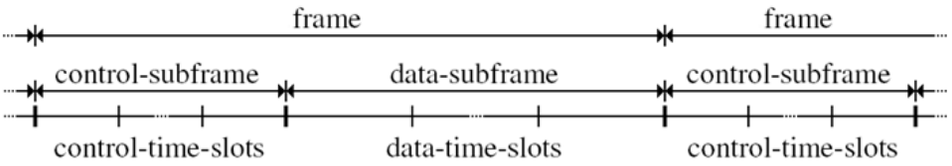

Figure 2. Frame Structure

Besides considering the interfering link sets of every link, the link-scheduling policy needs to consider the queue length of every link. In a wireless mesh network, when data packets are being transported over the flow's path, the links that form the path need to store the packets temporarily from the moment the node receives the packet until the moment the node forwards the packet to the next node in the path. Therefore, each link maintains queues of data packets for every flow that it belongs to. This is shown in Figure 3, which includes the queues of both link (i,j) and link (j,i). Each link has two queues. These are the input and output queues, which are denoted by $Q_i^{(i,j)}$ and $Q_o^{(i,j)}$ respectively for link (i,j). When node i receives a data packet that needs to be forwarded to node j, it stores the packet in $Q_i^{(i,j)}$ first. Then, it exchanges control packets with neighboring nodes in order to determine the data subframe and data-time slot when the data packet can be transmitted to node j without collisions. This is done according to the link-scheduling policy of the network. Once the transmission schedule of the data packet has been determined, the packet is moved to $Q_o^{(i,j)}$ where it waits for the data-time slot scheduled for its transmission. Finally, node i forwards the packet to node j at the scheduled data-time slot. At this point, the packet leaves $Q_o^{(i,j)}$. When node j receives the data packet, it checks whether it is the

packet's destination. If it is, the packet is no longer stored in any queue and leaves the network[2]. If it is not, it starts the link-scheduling process again in order to forward the packet to the next node in the data flow's path.

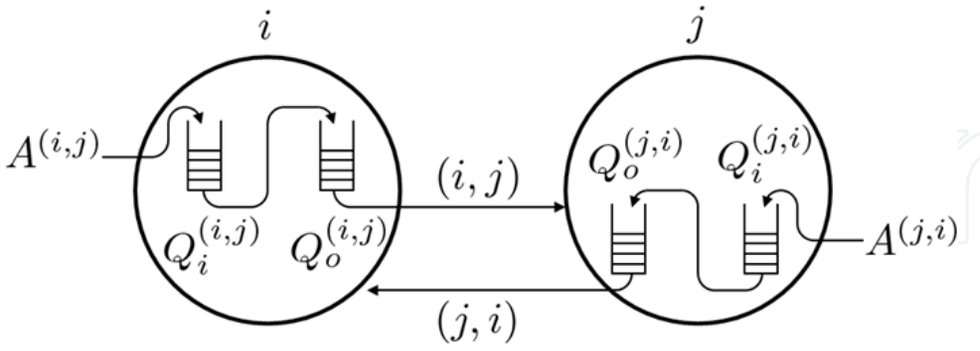

Figure 3. Data-packet transmissions over links (i,j) and (j,i)

When designing a link-scheduling policy, the goal is to support the largest set of data-packet rates for all the flows established in the network, and this should be done while guaranteeing the following conditions:

- There are no packet collisions
- The queues do not grow indefinitely
- A given level of fairness is guaranteed for all the flows

Packet collisions need to be avoided in order to guarantee the completeness of the information being delivered to the user. Given the limited amount of memory that nodes have, the queue lengths need to be guaranteed not to grow indefinitely. Otherwise, the nodes will drop data packets when they have run out of memory to store the packets while the transmission schedules are being determined. The fairness among data flows guarantees that each flow is assigned some part of the total capacity of the network to transport information[3].

The mathematical formulation of this problem is based on Markovian systems [9]. In order to do this formulation, the following definitions for each node's queues need to be considered first. In Equations 1 and 2, $\mathcal{S}_1^j$ is the set of 1-hop neighbors of node j. These are the nodes that have links with node j. Therefore, Q_i^j is the total number of packets stored in node j's 1-hop neighbors that need to be forwarded to node j and that are waiting to be scheduled, and Q_o^j is the maximum number of scheduled packets waiting to be forwarded to node j among all of j's 1-hop neighbors[4]. The time indexes n and m_n represent the n^{th}

[2]Actually, node j sends the data packet to its application layer so that the packet's content can be finally delivered to the user.

[3]It should be noted that there is not an absolute definition for fairness. For example, the network operator may be interested in assigning the same maximum data-packet rates to all flows or different maximum rates to different flows depending on the demands of the users.

[4]The actual length of Q_o^j has a more involved definition. However, for the sake of clarity, we do not consider the exact definition until Section 4.2.1.

time that at least one control packet is transmitted in the network and the control-time slot m_n when this takes place.

$$Q_i^j(n) \triangleq \sum_{i \in \mathcal{S}_1^j} Q_i^{(i,j)}(m_n) \tag{1}$$

$$Q_o^j(n) \triangleq \max_{i \in \mathcal{S}_1^j} \{Q_o^{(i,j)}(m_n)\} \tag{2}$$

Consider the following measure of the queue lengths of all links in the network, where $\mathcal{N}$ is the set of all nodes in the network, and the indexes i and o indicate whether input or output queues are being considered.

$$V_s^{i,o}(n) \triangleq \sum_{j \in \mathcal{N}} (Q_{i,o}^j(n))^2 \tag{3}$$

Intuitively, $V_s^{i,o}(n)$ can be interpreted as a total volume occupied by all of the input or output queues[5], depending on whether the index is i or o, and that is updated at every control-time slot in which there is at least one control-packet transmission. $V_s^{i,o}(n)$ increases and decreases randomly in time. It increases due to the data packets that the flows input into the network, and it decreases when data packets reach their destination and leave the network. This is shown graphically in Figure 4, which includes a network of 7 nodes. The volume of the network, shown in circles, increases and decreases according to the queue lengths in the network.

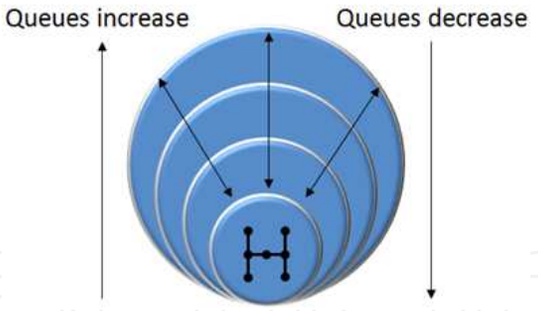

Figure 4. Network stability

Based on the concept of $V_s^{i,o}(n)$, the stability of a network can be defined as follows. A network is stable if $V_s^{i,o}(n)$ decreases to zero with some probability greater than zero at some finite future time $n+m$, i.e., there is a probability that the volume of the network decreases to zero within some finite time independently of the current volume. It can be shown that this condition is met if the expectation that $V_s^{i,o}(n)$ decreases is greater than zero [9]. Therefore, a network is stable if Equation 4 holds[6].

---

[5]In the theory of Markovian processes [9], $V_s^{i,o}(n)$ is known as a Lyapunov function.

[6] $\mathrm{E}[X \mid Y]$ denotes the expected value of X given Y.

$$\mathrm{E}[V_{\mathrm{s}}^{i,o}(n+1) - V_{\mathrm{s}}^{i,o}(n) \mid V_{\mathrm{s}}^{i,o}(n)] < 0 \tag{4}$$

A network becomes unstable when the rate at which data flows input packets into the network increases to a point in which the link-scheduling policy is not able decrease queue lengths fast enough to guarantee the condition given by Equation 4. Therefore, the task of the link-scheduling policy is to maintain the network stable under the constraints that there should not be data-packet collisions and that data-flows are fairly serviced.

The performance of the link-scheduling policy in performing this task is measured in terms of the set of data-packet rates for which it guarantees that the network is stable. The largest set of data-packet rates supported by the link-scheduling policy is known as the stability region. In order to compare different link-scheduling policies, these are usually compared against the optimal stability region, which is the largest region that any policy can achieve[7]. This comparison is done using the concept of efficiency ratio, which is defined as the fraction of the optimal stability region in which a suboptimal link-scheduling policy guarantees the stability of the network. Therefore, an optimal link-scheduling policy has an efficiency ratio of unity. When the link-scheduling policy has an optimal efficiency ratio, the network is able to support the largest set of data-packet rates, and so it achieves maximum throughput.

The stability region of most link-scheduling policies depends on the interference sets of the links in the network. This can be observed, for example, in the following case that considers two links of a network. If the two links interfere with each other, only one of them can be active at a time. However, if they do not interfere with each other, they can be active simultaneously. Therefore, when they do not interfere, the links are able to support higher data-packet rates for the flows that they belong to, and this increases the size of the stability region. Given that the interference sets are determined from the network topology, i.e., from the relative distance between nodes and their transmission powers, the stability region can be modified by controlling the network topology. Therefore, for a given network with a given link-scheduling policy and a given set of end-to-end data flows, the stability region can be adapted by means of topology control in order to increase the data-packet rates supported by the links for the flows that they belong to. An example of this adaptation is shown in Figure 5. This example considers two flows. There are an initial stability region and a final stability region. The coordinates of the operating point indicate the data-packet rates of the two flows. Therefore, as the flows increase their data-packet rates, the operating point moves further away from the origin. Given that the operating point has not crossed the boundary of the initial stability region, the network is stable. After controlling the network topology, the stability region is modified such that the distance from the boundary of the region to the operating point is increased. Therefore, the final stability region allows the operating point to be moved further away from the origin without crossing the boundary. In this way, the flows are able to operate at higher data-packet rates without destabilizing the network.

[7]The optimal stability region and the link-scheduling policy that achieves it were characterized in [10].

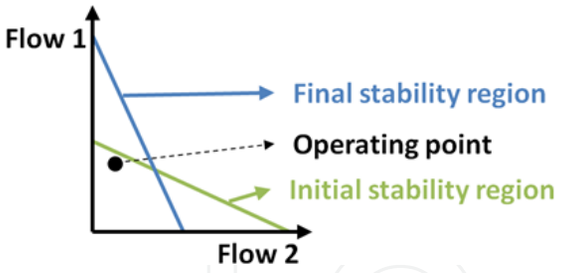

Figure 5. An example of stability-region adaptation

In the following, the operation and performance of the different link-scheduling policies is discussed. Special attention is given to reservation-based scheduling (RBDS) policies [8,11]. Then, based on the stability region of RBDS policies, the topology-control mechanisms are discussed.

2. Link-scheduling policies[8]

The challenge in link scheduling is that the policies are highly complex. The scheduling problem in general is nondeterministic polynomial time (NP) hard [12]. Therefore, the research literature has focused on policies of lower complexity that are more amenable to implementation [7].

Most distributed scheduling policies that achieve provable efficiency ratios calculate, at the onset of every frame, a subset of links that is allowed to transmit data in the immediately following frame only. In this chapter, we refer to these policies as non-RBDS policies, i.e., they do not reserve any future frame but only the following one. On the other hand, RBDS policies [8, 11] select links to transmit data in any future frames by means of frame reservations. Since this framework considers reservations of any future frames, non-RBDS policies correspond to a special case within the RBDS framework, i.e., the case that links are allowed to reserve the next frame only.

It should be noticed that non-RBDS policies require the input queue only (i.e., $Q_i^{(i,j)}$). They do not need the output queue (i.e., $Q_o^{(i,j)}$) because data packets do not need to wait for future data subframes. In non-RBDS policies, once a data packet is scheduled at the onset of the data subframe, the packet is transmitted immediately.

2.1. Non-RBDS policies

The concept of optimal stability region and a centralized scheduling policy with efficiency ratio of unity were introduced in [10]. The centralized scheduling policy attempts to solve a complex global optimization problem so that the entire network is stable for the largest possible set of input data-packet rates. Under the 1-hop interference model[9], the problem is

[8]The material presented in this section is based on the material presented in [8, 11].

[9]In the 1-hop interference model, only the links that the 1-hop neighbors of a node belong to interfere with the links that the node belongs to.

shown to correspond to a maximum weighted matching (MWM), where the weights of the links are determined from the length of their queues. The solution to MWM has complexity $O(N^3)$ [13, 14], where N is the number of nodes. Under the k-hop interference model, the problem has been proven to be NP-Hard [12]. Therefore, the optimal scheduling policy is not convenient for implementation due to its high complexity. As a consequence, less complex scheduling policies that achieve only a fraction of the optimal stability region for general network topologies have been developed [12, 15-28].

The different suboptimal scheduling policies proposed in the literature can be classified according to the techniques they use to calculate the next schedule. These techniques usually depend on the interference model assumed for the network and the links' weights at the onset of every frame. Also, the suboptimal scheduling policies can be further classified according to their centralized or distributed mechanism (Unless otherwise specified, the scheduling policies reviewed in this section consider 1-hop traffic only, i.e., the data flows' paths have one link only.).

2.1.1. Centralized policies

In [15], a centralized scheduling approach known as pick-and-compare [17] that achieves the optimal efficiency ratio is defined. The pick-and-compare scheduling policy selects the optimal schedule at every frame with some probability greater than zero. First, the scheduling algorithm randomly picks a new schedule such that the links can satisfy the interference model constraints. Then, the newly picked schedule is compared with the current schedule. If the picked schedule reduces the total weight of the network (i.e., queue lengths) more than the current schedule, then the picked schedule is selected as the next schedule; otherwise the current schedule is used again. The pick-and-compare policy requires the calculation and comparison of the updated total weight for every frame. Therefore, the complexity of this technique grows linearly with N, which makes it difficult to implement in networks with a high number of nodes or in networks where nodes have low processing capabilities.

Greedy maximal scheduling (GMS) is a suboptimal, centralized scheduling policy. In GMS, the links of the network are ordered according to their weights, where the link with maximum weight is placed at the top of this globally ordered list. A valid schedule is found by selecting links from the list from top to bottom that do not interfere with each other. The complexity of GMS is $O(L\log(N))$, where L is the number of links [29]. GMS has efficiency ratio of 1/2 under the 1-hop interference model [7], and under the k-hop interference model, GMS has efficiency ratio of 1, 1/6, and 1/49 for tree, geometric, and general network graphs respectively [12, 27].

2.1.2. Distributed policies

A distributed version of the pick-and-compare scheduling policy was proposed in [19]. In this policy, a node is selected with some probability less than one to initiate the calculation of a schedule for the links in its neighborhood. The new schedule is selected for the next

frame if the new schedule reduces the neighborhood's weight by more than the current schedule. The algorithm has constant complexity, so it does not depend on the number of nodes of the network. It does depend, however, on the diameter of the neighborhood. The efficiency ratio increases as the diameter of the neighborhood increases. The algorithm assumes the 1-hop interference model, so it can only be directly used on networks with physical layers such as frequency-hopping code-division-multiple-access (FH-CDMA) that allow that assumption to be made.

Greedy scheduling (GS) policies [29] have been developed that achieve the same efficiency ratio of GMS [17, 25, 28]. In the GS policies, nodes calculate locally the next schedule based on the links that have the maximum local weights.

In [17, 20-23], a maximal scheduling (MS) approach is described. In this approach, maximum weight is not required to schedule a link. A link is eligible for the next schedule as long as it has enough packets in the queue to transmit during the entire duration of a frame. The efficiency ratio of MS scheduling policies is $1/\kappa$, where κ is the maximum number of non-interfering links in the interference set of any link in the network. MS policies have also been adapted to multi-hop flow scenarios[10] in which a set of flows with their respective rates and routes are given [16, 20-23].

Lastly, distributed scheduling policies of complexity $O(1)$ have been developed in [18, 24, 30]. These are known as constant time (CT) scheduling policies [17]. The CT approach differs from the MS approach in that when a link does not interfere with the links in a schedule, it is selected with probability less than one. Therefore, in CT scheduling policies, frames can be wasted with some probability greater than zero. In [30], CT policies are proposed for the 1-hop and 2-hop interference models[11]. The efficiency ratios of these policies were improved in [18, 24]. In [25], the improved efficiency ratios are $\frac{1}{2}-\frac{1}{\sqrt{m}}$ and $\frac{2}{\hat{n}}\left(\frac{1}{2}-\frac{1}{\sqrt{m}}\right)$ for the 1-hop and 2-hop interference models respectively, where $\hat{n}$ is the maximum number of 1-hop neighboring links for any link of the network.

2.2. Reservation-based distributed scheduling

In an RBDS wireless network, the nodes negotiate with their neighbors the reservation of future data-time slots for their links. This negotiation is based on a three-way handshake that consists of a request, a grant, and a grant confirmation. Requests, grants, and grant confirmations are transmitted in scheduling packets. The nodes access the control-time slots for transmitting scheduling packets using an election algorithm. Therefore, in an RBDS wireless network, the nodes access the wireless channel using two different algorithms: the

[10]A multi-hop flow has a path that is at least 2 links long.

[11]In the 2-hop interference model, only the links that the 1-hop or 2-hop neighbors of a node belong to interfere with the links that the node belongs to. The 2-hop neighbors of a node are the nodes that have a shortest path to the node of length 2 links.

election algorithm and the RBDS algorithm, whose roles are to avoid collisions and wasted time slots in the control and data subframes respectively.

In this chapter we assume that the election algorithm is given, and we focus on the RBDS algorithm only. For example, in the IEEE 802.16 standard [31], the election algorithm is completely specified while the link-scheduling algorithm is not. The standard only specifies the control messages that can be used for the implementation of RBDS policies. We adopt the election algorithm of IEEE 802.16 wireless mesh networks with coordinated distributed scheduling. Also, it is assumed that the RBDS wireless network follows the 2-hop interference model, which is the model considered in the IEEE 802.16 standard [31].

In the IEEE-802.16 election algorithm, the nodes in every 2-hop neighborhood take turns by competing between them to access the control-time slots and transmit scheduling packets. Let the 2-hop neighborhood of node i, i.e., node i, its 1-hop neighbors, and its 2-hop neighbors, be denoted by $\mathcal{S}_{\leq 2}^{i}$. We model the operation of this election algorithm as follows[12].

- In order to avoid scheduling-packet collisions, no more than one node is selected in every $\mathcal{S}_{\leq 2}^{i}$ at any control-time slot.
- The nodes in $\mathcal{S}_{\leq 2}^{i}$, where i can be any node in $\mathcal{N}$, are selected in cycles. We refer to these cycles as scheduling cycles.
- Within a scheduling cycle, the nodes in $\mathcal{S}_{\leq 2}^{i}$ are selected once and only once each. The order in which they are selected is uniformly distributed among all the possible orders of selection.
- The order that nodes in $\mathcal{S}_{\leq 2}^{i}$ are selected is independent across scheduling cycles.

When nodes i and j exchange scheduling messages to perform the three-way handshake, they schedule data packets on link (i, j) and multicast the negotiated schedule to all links in $\mathcal{I}^{(i,j)}$. The handshake consists of the following steps[13].

1. Node i sends a request to node j for a certain number of data-time slots along with a set of data-time slot numbers that are available for reservation at node i.
2. Node j sends a grant to node i for the requested number of data-time slots according to its set of data-time slots available for reservation and those of node i.
3. Node i confirms the successful reception of the grant by echoing the grant in its next scheduling-packet transmission.

The reservation of the data-time slots takes place at steps 2 and 3. When node j transmits its scheduling packet, j's 1-hop neighbors receive the grant and mark the granted data-time slots as unavailable. When node i confirms the grant, i's 1-hop neighbors receive the grant and mark the granted data-time slots as unavailable too. Therefore, at the end of step 3, all

[12]The operation of the election algorithm for IEEE 802.16 mesh networks with coordinated distributed scheduling is described in detail in [32, 33].

[13]It is assumed that in this handshake node j grants node i's request and that the data-packet-slot reservation is successful at both i and j.

links in $\mathcal{I}^{(i,j)}$ have made the granted data-time slots unavailable (i.e., the grant has been multicast to all links in $\mathcal{I}^{(i,j)}$).

The requests and grants transmitted by the nodes are defined as follows.

Definition 1. Request $r_m^{(i,j)} \triangleq (f_s, f_x, z)$, where $(f_s, f_x, z) \in N^3$, is the request transmitted by node i at control-time slot m that requests for link (i,j) the data-time slots of z consecutive data-subframes starting at frame f_s or any other frame after f_s. Request $r_m^{(i,j)}$ expires at the onset of frame f_x.

Definition 2. Grant $g_m^{(i,j)} \triangleq (f_s, f_e)$, where $(f_s, f_e) \in \mathbf{N}^2$, is the grant transmitted by node j at control-time slot m that assigns to link (i,j) the data-time slots of the series of frames that starts and ends with frames f_s and f_e respectively. Grant $g_m^{(i,j)}$ expires at the end of frame f_e.

Definition 3. The length of grant $g_m^{(i,j)}$, denoted by $|g_m^{(i,j)}|$, is the number of data-subframes assigned in the grant. Therefore,

$$|g_m^{(i,j)}| \triangleq f_e - f_s + 1.$$

In order to implement RBDS policies, each node maintains two tables per link that the node belongs to. These are the unavailable-data-time-slots table and the requested-data-time-slots table. The tables are updated with the grants and requests exchanged with the node's 1-hop neighbors. An unavailable-data-time-slots table contains the set of unexpired grants that interfere with the link that the table belongs to. This set is denoted by $\mathcal{T}_u^{(i,j)}(m)$ for link (i,j) and is given by Equation 5, where $(g_m^{(x,y)})_{f_e}$ is the f_e component of $g_m^{(x,y)}$, and f_m is the current frame number (i.e., the frame that control-time slot m belongs to). The requested-data-time-slots table contains the set of unexpired requests made for the link the table belongs to. This set is denoted by $\mathcal{T}_r^{(i,j)}(m)$ for link (i,j) and is given by Equation 6, where $(r_m^{(i,j)})_{f_x}$ is the f_x component of $r_m^{(i,j)}$. $\mathcal{T}_u^{(i,j)}(m)$ and $\mathcal{T}_r^{(i,j)}(m)$ are functions of m given that the tables are updated with the grants and requests transmitted at every control-time slot.

$$\mathcal{T}_u^{(i,j)}(m) \triangleq \{g_l^{(x,y)} : (g_l^{(x,y)})_{f_e} \geq f_m, (x,y) \in \mathcal{I}^{(i,j)}, l \leq m\} \tag{5}$$

$$\mathcal{T}_r^{(i,j)}(m) \triangleq \{r_l^{(i,j)} : (r_l^{(i,j)})_{f_x} > f_m, l \leq m\} \tag{6}$$

In RBDS policies, two grants overlap with each other if the frame ranges given by their respective f_s and f_e frame numbers have one or more frame numbers in common.

2.2.1. RBDS Markovian system model

In order for RBDS policies to be mathematically characterized under the framework of network stability proposed in [10], it is necessary to show how networks that use RBDS policies can be modeled as Markovian systems [9].

In an RBDS network, each link has an input-queue and an output-queue as described in Section 3. The length of an input-queue (i.e., $Q_i^{(i,j)}(m)$) is defined as the number of data packets in the queue. The length of an output-queue (i.e., $Q_o^{(i,j)}(m)$) corresponds to the number of data-subframes in the following frame range: from the current frame to the last frame scheduled for the packets in the output-queue. Therefore, the length of output-queues does not depend on the number of scheduled packets waiting to be transmitted but on the schedules of those packets. The length of output-queues is given by Equation 7[14], where $\mathcal{T}_g^{(i,j)}$ is the set of unexpired grants of link (i,j) (i.e., $\mathcal{T}_g^{(i,j)}(m) \triangleq \{g_l^{(i,j)} : (g_l^{(i,j)})_{f_e} \geq f_m, l \leq m\}$).

$$Q_o^{(i,j)}(m) \triangleq [\max\left(\left\{(g)_{f_e} : g \in \mathcal{T}_g^{(i,j)}(m)\right\}\right) - f_m + 1]^+ \tag{7}$$

A node transmits scheduling packets by accessing control-time slots according to the election algorithm. The next control-time slot that node i is going to access is determined by this algorithm. This control-time slot is denoted by $M^i(m)$, i.e., at control-time slot m, the future control-time slot that node i transmits a scheduling packet is control-time slot $M^i(m)$.

Based on the previous definitions, RBDS wireless network $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ and $\mathcal{L}$ are the sets of nodes and links respectively, can be represented as a Markovian system whose state $\mathcal{S}^{\mathcal{G}}$ is given by the lengths of the input and output queues of all the links and the scheduling control-time slots of all the nodes. That is,

$$\mathcal{S}^{\mathcal{G}} \triangleq \left\{Q_i^{(i,j)}(m), Q_o^{(i,j)}(m), M^i(m) : (i,j) \in \mathcal{L}, i \in \mathcal{N}\right\}. \tag{8}$$

Within this framework for RBDS networks, the stability analysis of different RBDS policies can be performed. In the following, a greedy-maximal RBDS policy is considered.

2.2.2. The greedy-maximal RBDS policy and its stability region

The GM-RBDS policy is as follows. When any node i in $\mathcal{N}$ transmits a scheduling packet,

- It grants the longest request among all the unexpired requests made by its incoming links, and sets the grant's f_s component at the frame following the interfering grant that expires the latest.
- For every of its outgoing links, it requests as many consecutive data-subframes as unscheduled data packets cover entirely, sets every request's f_s component at the frame following the interfering grant that expires the latest, and sets every request's f_x component at the frame scheduled for its next scheduling-packet transmission.

When any node i in $\mathcal{N}$ receives a scheduling packet, it checks whether there is a grant in the packet and whether the grant is directed to one of its outgoing links. If that is the case, it confirms the grant only if the grant does not overlap with any of the grants in the link's unavailable-data-time-slots table.

---

[14] $[\cdot]^+$ is the positive-part operator.

The GM-RBDS policy is greedy maximal in the sense that the requests that are granted are the longest requests and each request corresponds to the maximum integer number of data-subframes that are covered by a link's unscheduled data packets (i.e., each request corresponds to $\left\lfloor \frac{Q_i^{(i,j)}}{m_{\text{ds}}} \right\rfloor$, where $Q_i^{(i,j)}$ is the number of unscheduled data packets to be transmitted on link (i,j), and m_{ds} is the number of data-time slots per data-subframe).

The size of the stability region of the GM-RBDS policy depends on the ability of the links to perform the three-way handshakes successfully. If the probability that a link finishes successfully a three-way handshake is low, the link's queue will decrease at a lower rate. Therefore, the link's ability to forward data packets within some time range is going to be lower (i.e., the highest packet rate supported by the link is lowered), and this reduces the size of the stability region. In [8], it was shown that the probability that a three-way handshake of link (i,j) is successful depends on the following aspects of the 2-hop neighborhoods of nodes i and j.

- The set of active nodes that i can listen to but j cannot, where an active node is a node that either forwards data-packets or is the destination node for at least one flow. This set is given by $\mathcal{S}_{\text{a}}^i \setminus \mathcal{S}_{\text{a}}^j$, where $\mathcal{S}_{\text{a}}^i$ is the set of active 1-hop neighbors of node i, and $\setminus$ refers to the relative complement, i.e., $\mathcal{S}_{\text{a}}^i \setminus \mathcal{S}_{\text{a}}^j \triangleq \{k \in \mathcal{S}_{\text{a}}^i : k \notin \mathcal{S}_{\text{a}}^j\}$.
- The degree $d^{(i,j)}$ of link (i,j), which is defined as the number flows that traverse link (i,j).
- The direct 1-hop neighborhood of a node which is defined as the set of 1-hop neighbors that send data packets to the node. Therefore, the direct 1-hop neighbors of a node always precede the node in at least one flow's path. Node j's direct 1-hop neighborhood is denoted by $\mathcal{S}_{\text{d}}^j$.

Based on the probability of successful three-way handshakes, sufficient conditions that guarantee queue stability under the GM-RBDS policy are given as follows[15].

Theorem 1. Let $\mathcal{G} = (\mathcal{N}, \mathcal{L})$ be a wireless mesh network that operates under GM-RBDS, shortest-path routing, and the 2-hop interference model, where $\mathcal{N}$ and $\mathcal{L}$ are the sets of nodes and links of the network respectively. Let λ_{f}^j be the maximum packet rate that node j can support for each of the flows for which it is an intermediate or destination node. $\mathcal{G}$ is stable if the packet rate λ_{f}^j supported by every node j in $\mathcal{N}$ satisfies Equation 9.

$$\lambda_{\text{f}}^j < \frac{1}{5 \sum_{i \in \mathcal{S}_{\text{d}}^j} d^{(i,j)} \, | \mathcal{S}_{\text{a}}^i \setminus \mathcal{S}_{\text{a}}^j |} \quad \forall \; j \in \mathcal{N} \tag{9}$$

[15]The proof of Theorem 1 is given in [34].

Therefore, in order to guarantee stability under shortest-path routing and GM-RBDS, the data-packet rate of a flow must be less than the following rate: the minimum packet rate among all the packet rates that nodes along the flow's path can assign to the flow. This is shown in Equation 10, where $\mathcal{F}$ is the set of data flows in the network, f_n is the n^{th} flow in $\mathcal{F}$, p_n is the path followed by f_n, λ_n is the data-packet rate of flow f_n, and $\lambda_{\max}^j$ is the upper-bound for node j's rate λ_{f}^j according to Equation 9 (i.e., $\lambda_{\max}^j \triangleq (5\sum_{i\in\mathcal{S}_{\text{d}}^j} d^{(i,j)} \mid \mathcal{S}_{\text{a}}^i \setminus \mathcal{S}_{\text{a}}^j \mid)^{-1}$).

$$\lambda_n < \min\{\lambda_{\max}^j : j \in p_n\} \quad \forall \ f_n \in \mathcal{F} \tag{10}$$

Remark. Notice that the sufficient condition for stability given by Equation 9 is of the same form of the condition for the non-RBDS greedy policies analyzed in [23] (Equation 4 in [23]). That is, the total packet-arrival rate of a set of interfering links needs to be lower than some constant in order to guarantee stability, and the constant depends on some characteristic of the network topology (i.e., $\mathcal{S}_{\text{a}}^i \setminus \mathcal{S}_{\text{a}}^j$ for the GM-RBDS policy, and κ for the greedy policies in [23]). Other policies have the same behavior as well. For example, the stability properties of GMS [27] and the bipartite simulation (BP-SIM) [24] policies depend on the local-pooling factor[16] and the maximum node degree[17] of the network respectively, and these are determined by the network topology.

3. Stability-based topology control[18]

In this section, we look at the problem of topology control for adapting the stability region of the backbone of the wireless mesh network to a given set of flows such that the total throughput is improved. This topology-control framework was originally studied in [34-36]. Specifically, we ask the question of what are the nodes' transmission powers (TP) that adapt the stability region to the flows in the network when a set of source-destination pairs, the routing algorithm, and the link-scheduling policy are given. Notice that by adapting the TPs of the nodes (i.e., wireless mesh routers and gateways), the topology of the network is being controlled due to the creation and elimination of links. Also, notice that the flows correspond to the traffic established across the wireless mesh routers and gateways of the network.

By adapting the stability region of the network, the queue lengths across the network are decreased in average for a given set of flows' data-packet rates. In this way, the flows among the source-destination pairs are able to maintain higher levels of end-to-end throughput and lower levels of end-to-end delay while guaranteeing queue stability. Therefore, the problem considered in this chapter is of particular interest for applications that establish non-bursty sessions between source-destination pairs such as audio/video calls.

---

[16] The local-pooling factor is a topological property of the network that indicates how different the effectiveness of the different maximal link schedules is from each other [27]. When the different maximal link schedules are similarly effective, GMS policies are able to support packet rates that are closer to the boundaries of the optimal stability region.

[17] The node degree is defined as the number of links that the node belongs to.

[18] The material presented in this section is based on the material presented in [34, 36].

In order to adapt the stability region, we propose an algorithm that is executed by the flows established between the source-destination pairs. The idea behind the algorithm is to adapt a lower-bound region of the stability region (i.e., a region covered by the stability region) by modifying the TPs. The lower-bound region is a widely accepted theoretical performance metric used for comparing different link-scheduling policies [23][19]. In the algorithm, once the flows' paths are determined by the routing algorithm, the flows calculate the maximum data-packet rate they can support within the lower-bound region; then, each flow tries to stretch the lower-bound region by modifying the TP of nodes surrounding it. The effect that the stretch of the lower-bound region has on the stability region is another stretch on this region. Therefore, the result is a stability region adapted to the flows that allows them to support higher data-packet rates while guaranteeing the stability of the network. A graphical example of this adaptation was shown in Figure 5.

We consider IEEE 802.16 wireless mesh networks that operate under shortest-path routing and the GM-RBDS policy. However, our results can be readily extended to other networks, routing algorithms, and link-scheduling policies.

3.1. Stability-region expansion algorithms

The main idea presented in this chapter (i.e., adapting the stability region of a given link-scheduling policy by means of TP control) is based on the results obtained in [37, 38, 39]. In [37], the network is partitioned based on the notion of local pooling, and each partition is assigned to a channel of the network. In this way, the GMS policy is guaranteed to achieve the optimal stability region in each channel. In [38, 39], network topologies are identified for which distributed link-scheduling policies achieve the optimal stability region. However, these network topologies are not suitable for real scenarios [27] because of their sufficient conditions that guarantee the optimal stability region. These conditions include [38] 1-hop interference, 1-hop traffic, and a topology that is a graph that belongs to one of the following perfect-graph classes: chordal graphs, chordal bipartite graphs, cographs, and a subgroup of co-comparability graphs. In real scenarios, these conditions limit the suitability of wireless mesh networks. For example, only a few physical-layer technologies such as code-division-multiple-access (CDMA) can be approximated with the 1-hop interference model, and the traffic in wireless mesh networks is multihop by definition. Also, making the topology fall within the previous graph families imposes constraints on the locations and TPs of the nodes and the available routes. The multihop traffic case was considered in [38], and it was shown that only a subset of the previous graph families guarantee the optimal stability region in the multihop-traffic scenario. These were identified as forest of stars, where every connected component of the network graph is a star graph. Also, the results in [37, 38] are valid only for GMS policies under 1-hop traffic or backpressure routing-scheduling policies under multihop traffic[20]. In

[19]The reason for this is that the exact formulation of the stability region is not actually available. The stability region is usually characterized with the lower-bound region because its exact characterization is not feasible due to its complexity. See [8, 10, 11, 16, 18, 19, 23-25, 29, 30] for the literature on the problem of characterizing the stability region of link-scheduling policies.

[20]It should be noted that the objective in [37, 38] was mainly to identify the topologies that enable the optimality of the GMS policy, and not to design topology-control algorithms.

[40], a random-power-selection algorithm for random-access scheduling policies was proposed. It is shown that it achieves maximal throughput in the following sense: the throughput achieved by any fixed power selection is at most equal to the throughput achieved by the random-power-selection algorithm.

Our approach is built upon the idea of [37, 38] that under certain topologies a link scheduling policy performs better. We modify realistically the network topology using TP control to adapt the policy's stability region to the flows. The algorithm receives any set of end-to-end paths, node locations, and scheduling policy, and adapts the policy's stability region to the paths. Such an approach is beneficial because it improves the end-to-end throughput and delay without the restrictions previously discussed. In this chapter, we consider the case of shortest-path routing, GM-RBDS scheduling, and randomly chosen source-destination pairs of nodes in IEEE 802.16 mesh networks.

Other heuristic algorithms have been proposed in the literature that improve the performance of the link-scheduling policy in terms of throughput by means of TP control. These algorithms include the ones reported in [41-43] whose basic idea is to increase the total throughput in the network by means of spatial reuse. The spatial reuse is increased by reducing the TP of the nodes. The algorithms differ between them in the way they are adapted to request-to-send (RTS)/clear-to-send (CTS) based protocols. In [44, 45], it is shown that better throughput improvements can be achieved not only by decreasing the TP to increase the spatial reuse but also by considering the hidden and exposed nodes. The algorithms proposed in [44] perform TP control with the objective of avoiding hidden nodes. In this way, the links in the network are able to sustain higher data-packet rates. In [46], a TP control algorithm for RTS/CTS-based protocols is proposed that decreases the area occupied by links during their transmissions, which is defined as the area in which other nodes must remain silent during the time the link is active. Then, it is shown that with this scheme, routing algorithms that favor short hops achieve higher levels of throughput. The goal of our algorithm is similar to the goal of the previous algorithms [41-46], i.e., to increase the data-packet rates that a given link-scheduling policy can support by means of TP control. However, our approach differs in that it is directly based on a quantitative metric which is the stability region. It is not based on qualitative observations of the operation of the link-scheduling policy such as the hidden and exposed nodes in RTS/CTS-based policies. Therefore, it can be readily adapted to any link-scheduling policy whose stability region has been characterized such as the ones discussed in Section 4.

A different type of TP control algorithms, which are based on optimization techniques, are discussed in [47, 48]. In [47], the problem of integrated link scheduling and TP control for throughput optimization is shown to be nondeterministic polynomial time (NP) complete. Therefore, a heuristic algorithm is developed. The goal of the algorithm is to minimize the schedule length necessary to satisfy all the link loads determined by a given routing algorithm. By minimizing the schedule length, the total throughput of the network is increased because more scheduling cycles can be performed per time unit. In [48], the problem of jointly optimizing the flow routes, link schedules, TP, modulation and coding schemes is addressed. This is a more general problem than the one considered in [47] given that it does

not only include the calculation of TPs and link schedules but also includes the routing and physical layers (i.e., flow routes, modulation, and coding schemes). In our algorithm, we are only concerned in the TP control problem when the flows and link-scheduling policy are given. That is, for a given set of flows, we determine TPs that improve the performance of the link-scheduling policy in terms of throughput and end-to-end delay.

3.2. The HSRA-topology-control algorithm

The goal of our TP control algorithm is to expand the lower-bound region given by Equation 10. By expanding this region, the flow rates λ_n can take higher values while guaranteeing stability, and therefore, the maximum total throughput the network can support for the given flows is increased. Let the maximum total throughput be denoted by λ_{T} and defined in terms of the lower-bound region for the flows' data-packet rates given by Equation 10 as follows.

$$\lambda_{\mathrm{T}} \triangleq \sum_{p_n \in \mathcal{F}} \min\{\lambda_{\max}^{j} : j \in p_n\} \tag{11}$$

According to Equations 11, 10, and 9, λ_{T} depends on the direct 1-hop neighborhoods (i.e., $\{\mathcal{S}_{\mathrm{d}}^{j} : j \in \mathcal{N}\}$), the link degrees (i.e., $\{d^{(i,j)} : (i,j) \in \mathcal{L}\}$), and the active 1-hop neighborhoods (i.e., $\{\mathcal{S}_{\mathrm{a}}^{j} : j \in \mathcal{N}\}$) as follows.

$$\lambda_{\mathrm{T}} = \sum_{p_n \in \mathcal{F}} \left(\min_{j \in p_n} \frac{1}{5 \sum_{i \in \mathcal{S}_{\mathrm{d}}^{j}} d^{(i,j)} \mid \mathcal{S}_{\mathrm{a}}^{i} \setminus \mathcal{S}_{\mathrm{a}}^{j} \mid} \right) \tag{12}$$

Given that the flows are determined by the shortest-path routing algorithm, the following parameters in Equation 12 are fixed: $\{p_n \in \mathcal{F}\}$, $\{\mathcal{S}_{\mathrm{d}}^{j} : j \in \mathcal{N}\}$, and $\{d^{(i,j)} : (i,j) \in \mathcal{L}\}$. Therefore, in order to increase λ_{T}, the only parameters that can be modified are the active 1-hop neighborhoods (i.e., $\{\mathcal{S}_{\mathrm{a}}^{j} : j \in \mathcal{N}\}$). They can be modified by means of TP control such that λ_{T} is maximized. This optimization problem, which we call stability region adaptation for throughput maximization (SRA-TM), is given as follows.

Definition 4. Given a set of flows $\mathcal{F}$ calculated by the shortest-path routing algorithm, the SRA-TM problem consists of the maximization of λ_{T} by means of TP control such that none of the nodes exceed the maximum TP and none of the paths are broken. That is,

$$\begin{aligned} \text{maximize} \quad & \sum_{p_n \in \mathcal{F}} \left(\min_{j \in p_n} \frac{1}{5 \sum_{i \in \mathcal{S}_{\mathrm{d}}^{j}} d^{(i,j)} \mid \mathcal{S}_{\mathrm{a}}^{i} \setminus \mathcal{S}_{\mathrm{a}}^{j} \mid} \right) \\ \text{subject to} \quad & 0 \le r^{i} \le r^{\max} \quad \forall \ i \in \mathcal{N} \\ & r^{i}, r^{j} \ge \mid i, j \mid \mid \quad \forall \ i \in \mathcal{S}_{\mathrm{d}}^{j}, j \in \mathcal{N}, \end{aligned} \tag{13}$$

where r^i is the transmission radius of node i, $r^{\max}$ is the maximum transmission radius, and $||i,j||$ is the Euclidean distance between nodes i and j.

Remark. In the SRA-TM problem, the flow paths are given and left unmodified. Higher values for λ_T could be achieved if the flow paths were modified by including them as decision variables. For example, a routing scheme can uniformly distribute the traffic loads across the links of the network so that links with high levels of congestion are avoided. This problem corresponds to a joint optimization of the topology and flow paths based on the stability region. This problem can be further studied due to its potential benefits on λ_T. However, this chapter deals only with the stability-region-based topology control as a first step towards the problem of stability-region-based joint topology and routing control.

Remark. If the data traffic in the network changes dynamically, the flow paths may change as well. In this scenario, the SRA-TM problem needs to be solved for every flow-path change. Therefore, the speed of convergence of algorithms that solve the SRA-TM problem is an important metric for such a scenario. The algorithms should be able to keep up with the rate of change of the flow paths. On the other hand, if the data-traffic levels of a set of flows change but the flow paths do not change, the SRA-TM problem does not need to be solved again. The reason is that the solution of the SRA-TM problem is the topology that allows those flows to support the maximum level of data traffic while guaranteeing stability. This means that the data-traffic levels in the flows may vary as long as they do not exceed such maximum levels (i.e., $\min\{\lambda^j_{\max} : j \in p_n\} \quad \forall \ f_n \in \mathcal{F}$), and this can be guaranteed by means of call-admission-control algorithms.

In order to solve the SRA-TM problem, the following TP algorithm is proposed. It is called heuristic stability region adaptation (HSRA)[21].

The following definitions are necessary for the operation of the HSRA algorithm.

Definition 5. The bottleneck node of flow f_n is the node with the lowest maximum rate among all the intermediate and destination nodes of the flow, i.e., let j be the bottleneck node of f_n, then $j = argmin_{i \in \{p_n(m):2 \le m \le |p_n|\}} \lambda^i_f$, where p_n is the set of nodes in the path of flow f_n, $p_n(m)$ is the m^{th} node in p_n, and $|p_n|$ is the number of nodes in p_n.

Definition 6. Node h is hidden from node j if and only if $h \in \mathcal{S}^i_a \setminus \mathcal{S}^j_a$ for some $i \in \mathcal{S}^j_d$.

Definition 7. The MinPower setup is the set of minimum TPs whose transmission ranges guarantee that none of the links of the flows in $\mathcal{F}$ is broken.

The operation of the algorithm is as follows. First, the nodes' TPs (i.e., $\{r^i : i \in \mathcal{N}\}$) are set according to the MinPower setup (line 2 of the HSRA Algorithm). By reducing the TPs (Definition 7), the spatial reuse in the network is increased, and as a consequence, the total throughput is increased as well[22]. Then, the maximum throughput that intermediate and

[21]The SRA-TM problem is formulated as a mixed integer program with non-linear constraints in [34]. This formulation is used in Section 6 for calculating the optimal solution of the simulated instances of the SRA-TM problem.

[22]This spatial-reuse-based TP control is the basis of the algorithms proposed [39-43, 46].

HSRA Algorithm

```
1: procedure HSRA(𝒩, ℱ, M)
2:     {r^i} ← MinPowerSetup(𝒩)
3:     {λ^i_max} ← NodeMaxRates(𝒩)
4:     T ← TotalThroughput(ℱ, {λ^i_max})
5:     for m ← 0, m < M, m ← m + 1 do
6:         f_n ← PickFlowRandomly(ℱ)
7:         j ← BottleneckNode(f_n, {λ^i_max})
8:         𝒮^j_d ← Direct1HopNeigh(j)
9:         for every i in 𝒮^j_d do
10:            𝒮^i_a \ 𝒮^j_a ← HiddenActiveNodes((i, j))
11:        end for
12:        i ← NodeHiddenTheMost({𝒮^i_a \ 𝒮^j_a})
13:        if ||i, j|| < r^max then
14:            r_aux ← r^i
15:            r^i ← ||i, j||
16:            {λ^i_aux} ← NodeMaxRates(𝒩)
17:            T_aux ← TotalThroughput(ℱ, {λ^i_aux})
18:            if T_aux > T then
19:                {λ^i_max} ← {λ^i_aux}
20:                T ← T_aux
21:            else
22:                r^i ← r_aux
23:            end if
24:        end if
25:    end for
26: end procedure
```

destination nodes can support for the flows they belong to is calculated (line 3 of the HSRA Algorithm). This is done using Equation 9, which defines the nodes' maximum throughput. Based on these maximums, the total throughput the network can support is calculated (line 4 of the HSRA Algorithm).

Once the total throughput under the MinPower setup is known, flows are selected randomly one-by-one for a number of M times (line 5 of the HSRA Algorithm). Every time a flow is selected, the maximum throughput the flow can support is increased if this causes that the total throughput be increased as well. Otherwise, the flow is left unmodified. The throughput of the selected flow is increased as follows.

Let the selected flow be denoted by f_n (line 6 of the HSRA Algorithm). The bottleneck node of f_n is found first by tracking the node of the flow with the lowest maximum throughput (Equation 10). Let this node be denoted by j (line 7 of the HSRA Algorithm). The maximum rate of j (i.e., λ_f^j) is increased by increasing the TP of one of j's 2-hop neighbors (lines 8 to 15 of the HSRA Algorithm). However, this TP increase is confirmed only if the total throughput (i.e., λ_T) is increased as well (lines 16 to 20 of the HSRA Algorithm). Otherwise, the TP of j's 2-hop neighbor is left unmodified (line 22 of the HSRA Algorithm). The total throughput may be decreased given that the TP increase of j's 2-hop neighbor may

decrease the maximum rate of other bottleneck nodes in the network, and this maximum-rate decrease may be higher than the increase on j's maximum rate.

The 2-hop neighbor of node j whose TP is increased is selected so that the factor $|\mathcal{S}_{\mathrm{a}}^{i} \setminus \mathcal{S}_{\mathrm{a}}^{j}|$ on the denominator of the upper-bound for λ_{f}^{j} is decreased (Equation 9). Qualitatively, this TP increase can be explained as follows. Node j (i.e., the bottleneck node) has a set of 1-hop neighbors that are sending data packets to it (i.e., $\mathcal{S}_{\mathrm{d}}^{j}$). Let i be one of these nodes, and consider the link (i,j) and the input and output queues $Q_i^{(i,j)}$ and $Q_o^{(i,j)}$ of node i as shown in Figure 3. In order for i to transmit packets to j, a reservation of future data-time slots is required. When nodes i and j finish this reservation successfully, data packets in node i's input-queue (i.e., $Q_i^{(i,j)}$) are moved to node i's output-queue (i.e., $Q_o^{(i,j)}$), and these packets are later pulled from $Q_o^{(i,j)}$ for their transmission. Therefore, for the queues $Q_i^{(i,j)}$ and $Q_o^{(i,j)}$ to have their lengths decreased, the reservation performed by nodes i and j needs to be successful, i.e., the three-way handshake for scheduling data-packet transmissions on link (i,j) needs to be successful. The probability that the handshake is successful and that the queues decrease their length depends on the grants received by node i and not received by node j. In the following, we refer to these grants as hidden-grants. If i requests future data-time slots to j and a hidden grant is received by i before j transmits its grant to i, j's grant may not be confirmed by i. This is because the hidden grant may interfere with j's grant. On the other hand, if j is able to listen to the hidden grant, j is able to generate its grant such that it does not interfere with the hidden grant, and i will be able to confirm j's grant[23]. Therefore, in order to increase the probability of handshake success and queue decrease, the TP of the node that transmits the hidden grant (i.e., the node hidden from j) can be increased such that node j is able to listen to the hidden node's transmissions.

Node j may have more than 1 hidden nodes in every incoming link from the nodes in its direct 1-hop neighborhood. The HSRA algorithm chooses only one of those hidden nodes for increasing its TP. The node that is chosen is the node that is hidden from the highest number of nodes (i.e., node j and all the other intermediate or destination nodes unable to listen to the hidden node). This is performed in lines 8 to 12 of the HSRA Algorithm. In this way, the maximum rate is increased for all those nodes so that, if one or more of those nodes are bottleneck nodes, higher improvements on the total throughput can be achieved.

The role that the objective function of the SRA-TM problem (Equation 13) plays in the HSRA Algorithm is the quantification of the throughput improvement by the TP increase on hidden nodes. By increasing the TP of a node hidden from a bottleneck node, the factor $\mathcal{S}_{\mathrm{a}}^{i} \setminus \mathcal{S}_{\mathrm{a}}^{j}$ in the denominator of Equation 13 is decreased for the bottleneck node, and as a consequence the bottleneck node's maximum rate is increased. However, the TP increase on the hidden node may also cause an increase on the $\mathcal{S}_{\mathrm{a}}^{i} \setminus \mathcal{S}_{\mathrm{a}}^{j}$ factor of other bottleneck nodes. Therefore, the objective function allows the algorithm to trade off between decreasing the number of hidden nodes by increasing TP and maintaining spatial reuse by not increasing

[23]The problem of node j not being able to listen to hidden grants is the hidden-node problem version for reservation-based distributed scheduling policies. This problem is studied in detail in [8, 11].

TP. In the algorithm, this tradeoff is achieved by testing the improvement on the total throughput (lines 14 to 23 of the HSRA Algorithm).

4. Simulation results

The performance evaluation of the HSRA algorithm was performed by means of simulation using the simulator proposed in [49]. The simulated network is an IEEE 802.16 mesh network with distributed scheduling under the configuration shown in Table 1. The number of nodes was specified as a simulation parameter. The nodes were uniformly distributed in a square area such that the node density was always kept at 15 nodes per unit area. The maximum transmission range of the nodes was set at 0.3 (i.e., $r^{\max} = 0.3$). The connectivity of the network under bidirectional links and with the nodes' transmission ranges set at $r^{\max}$ was confirmed before executing the shortest-path routing algorithm. The number of flows was specified as a simulation parameter. The source and destination nodes of every flow were uniformly distributed among all the nodes in the network. The shortest-path routing algorithm calculated the flow paths under the MaxPower setup which is the power assignment when all the nodes' transmission ranges are set at the maximum (i.e., $r^{\max}$). Once the paths were calculated, the transmission ranges of the nodes were found using the HSRA algorithm. Also, the optimal transmission ranges (i.e., the solution to the SRA-TM problem (Equation 13)), which we call OptPower, were found in [34] using the formulation of the SRA-TM problem as a mixed integer program with non-linear constraints (MIP-NLC). The MIP-NLC was solved using the Branch And Reduce Optimization Navigator (BARON) Solver [50], which is a system for solving non-convex optimization problems to global optimality. Finally, the network was simulated under the MaxPower, MinPower, OptPower, and HSRA setups.

Parameter	Value
Frame length	10 ms
Control-time-slot length	63 μs
Number of control-time slots per frame	4
Number of data-time slots per frame	256
NextXmtMx †	7
XmtHoldoffExponent †	6
Link scheduling	GM-RBDS
Routing	shortest-path

Table 1. IEEE 808.16 mesh network configuration

[†] This is a parameter of the election algorithm used t specify the frequency that nodes transmit scheduling packets.

Figure 6 shows the average output-queue length for three networks with 20 nodes each and increasing number of flows (i.e., 10 flows in Figure 6a, 15 flows in Figure 6b, and 20 flows in

Figure 6c). The input-queues have been omitted because they are guaranteed to always be stable [8, 11]. The flow rates in each network were all set at the same value. These are 8, 6, and 5 packets per frame for the networks in Figures 6a, 6b, and 6c respectively. These values were set so that their corresponding HSRA network became unstable if they were increased by at least one point. In this way, the network operates at a point inside the stability region and close to its boundary. Therefore, when any of the rates is increased by at least one point, the network operates outside the stability region, and therefore, it is unstable.

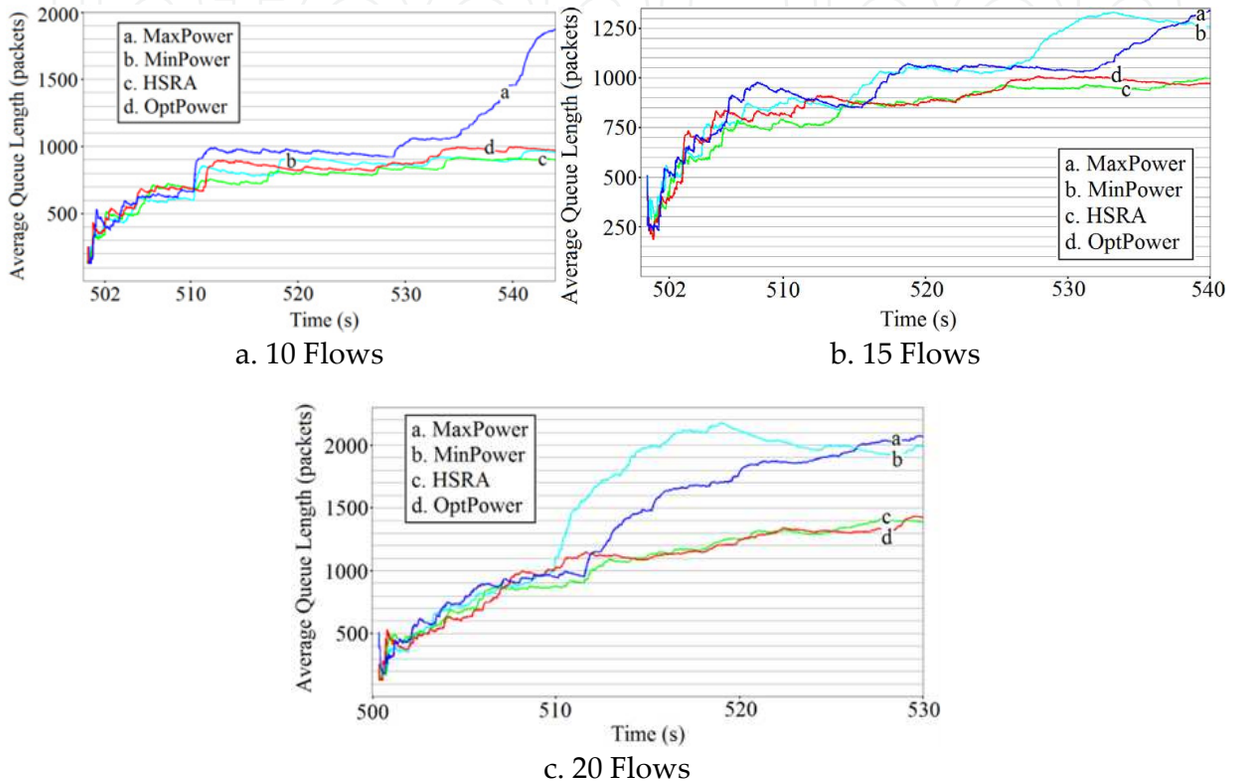

Figure 6. Average output-queue comparison for the HSRA, MinPower, MaxPower, and OptPower configurations

Close to the end of the simulation time, when the transient behavior of the queues is over, the average queue lengths of the different power setups (i.e., MaxPower, MinPower, OptPower, and HSRA) can be compared. In Figure 6a, the MaxPower setup has the worst performance (i.e., the largest average queue length), and the MinPower, OptPower, and HSRA have similar performance. Therefore, when the number of flows is low (i.e., 10 flows), the MinPower and the HSRA algorithms are able to achieve queue lengths that are close to the lengths achieved by the optimal solution (i.e., OptPower). On the other hand, when the number of flows increases, the MinPower algorithm does not achieve a performance close to the optimal one while the HSRA algorithm does. This is shown in Figures 6b and 6c. The MaxPower and MinPower algorithms have similar performance which is worse when compared with the HSRA algorithm. The HSRA algorithm achieves average queue lengths that are close to the lengths achieved by the optimal solution. Therefore, the HSRA algorithm enables the flows to carry more traffic while guaranteeing stability than the MaxPower and

MinPower algorithms do. Also, it is confirmed that the technique of only maximizing the spatial reuse by reducing the transmission ranges (i.e., MinPower) does not perform well when the flow density increases (i.e., when the number of flows increases and the number of nodes is kept constant.). On the other hand, the technique of adapting the stability region to the given set of flows by means of TP control (i.e., HSRA) does perform well when the flow density increases.

5. Conclusion

A new framework for the stability analysis of scheduling policies for wireless networks that allow the reservation of future data-subframes has been proposed. The concepts of input-queue and output-queue were introduced into the framework in order to account for the packets waiting to be scheduled and the schedules assigned to these packets. Based on these concepts, sufficient conditions for the stability of RBDS wireless networks were found.

Within the proposed framework, an RBDS policy which uses the concept of greedy-maximal scheduling was analyzed. The nodes implement this policy by exchanging scheduling packets using the IEEE 802.16 election algorithm. A region in which the proposed reservation-based scheduling policy is stable was found using the framework. It was shown that the size of this region depends on a characteristic of the network topology (i.e., Q_o^j).

The HSRA algorithm has been proposed for transmission power control. This algorithm increases the data-packet rates that flows can support and decreases the end-to-end delays. It is based on the adaptation of the stability region of a given link-scheduling policy when only the links that belong to a given set of flows are considered. The algorithm can be readily adapted to any link-scheduling policy whose stability region has been characterized, so it is not limited to any specific scheduling approach such as RTS/CTS-based policies. The improvement on throughput achieved by our algorithm was evaluated by means of. It was shown that it outperforms the classical solution of reducing transmission powers to increase spatial reuse.

Future lines of research include the development of a new framework for distributed topology-control algorithms. For example, this framework could based on a game-theoretical approach in which a given set of flows act as players that collaborate to maximize the packet rates they can support while guaranteeing stability. Also, based on the new framework, new distributed topology-control algorithms should be developed for IEEE 802.16 WMNs. Finally, the algorithms should be implemented and tested on WMN testbeds in order to evaluate the improvement in throughput they achieve in a real scenario.

Author details

Gustavo Vejarano
Department of Electrical Engineering and Computer Science,
Loyola Marymount University, Los Angeles, CA, USA

6. References

[1] Kawadia, V. and Kumar, P. R. (2005) Principles and protocols for power control in wireless ad hoc networks. IEEE Journal on Selected Areas in Communications. 23: 76–88.

[2] Santi, P. and Blough, D. M. (2003) The critical transmitting range for connectivity in sparse wireless ad hoc networks. IEEE Transactions on Mobile Computing. 2: 25–39.

[3] Sridhar, A. and Ephremides, A. (2008) Energy optimization in wireless broadcasting through power control. Ad Hoc Networks. 6: 155–167.

[4] Chiang, M., Tan, C.W., Palomar, D. P., O'Neill, D., and Julian, D. (2007) Power control by geometric programming. IEEE Transactions on Wireless Communications. 6: 2640–2651.

[5] Kim, T.-S., Lim, H., and Hou, J. C. (2008) Understanding and improving the spatial reuse in multihop wireless networks. IEEE Transactions on Mobile Computing. 7: 1200–1212.

[6] Chiang, M. (2005) Balancing transport and physical layers in wireless multihop networks: Jointly optimal congestion control and power control. IEEE Journal on Selected Areas in Communications. 23: 104–116.

[7] Lin, X. and Shroff, N. (2006) The impact of imperfect scheduling on cross-layer congestion control in wireless networks. IEEE/ACM Transactions on Networking. 14: 302–315.

[8] Vejarano, G. and McNair, J. (2012) Stability analysis of reservation-based scheduling policies in wireless networks. IEEE Transactions on Parallel and Distributed Systems. 23: 760–767.

[9] Asmussen, S. (2003) Applied Probability and Queues. New York: Springer. 438 p.

[10] Tassiulas, L. and Ephremides, A. (1992) Stability properties of constrained queuing systems and scheduling policies for maximum throughput in multihop radio networks. IEEE Transactions Automatic Control. 37: 1936–1948.

[11] Vejarano, G. and McNair, J. (2010) Reservation-based distributed scheduling in wireless networks. Proc. 11th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'10). Montreal, QC, Canada. pp. 1–9.

[12] Sharma, G., Mazumdar, R., and Shroff, N. (2006) On the complexity of scheduling in wireless networks. Proc. Annual International Conference on Mobile Computing and Networking (MobiCom'06). Los Angeles, CA. pp. 227–238.

[13] Hajek, B. and Sasaki, G. (1988) Link scheduling in polynomial time. IEEE Transactions on Information Theory. 34: 910–917.

[14] Papadimitriou, C. and Steiglitz, K. (1982) Combinatorial Optimization: Algorithms and Complexity. Prentice-Hall.

[15] Tassiulas, L. (1998) Linear complexity algorithms for maximum throughput in radio networks and input queued switches. Proc. Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'98). San Francisco, CA. pp. 533–539.

[16] Wu, X. and Srikant, R. (2005) Regulated maximal matching: A distributed scheduling algorithm for multi-hop wireless networks with node-exclusive spectrum sharing. Proc. IEEE Conference on Decision and Control, and the European Control Conference (CDC-ECC'05). Seville, Spain. pp. 5342–5347.

[17] Sharma, G., Joo, C., and Shroff, N. (2006) Distributed scheduling schemes for throughput guarantees in wireless networks. Proc. Annual Allerton Conference on Communication, Control, and Computing (Allerton'06). Monticello, IL. pp. 1–10.

[18] Joo, C. and Shroff, N. (1997) Performance of random access scheduling schemes in multihop wireless networks. Proc. Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'97). Kobe, Japan. pp. 19–27.

[19] Sanghavi, S., Bui, L., and Srikant, R. (2007) Distributed link scheduling with constant overhead. Proc. International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS'07). San Diego, CA. pp. 313–324.

[20] Chaporkar, P., Kar, K., and Sarkar, S. (2005) Throughput guarantees through maximal scheduling in multi-hop wireless networks. Proc. 43rd Annual Allerton Conference on Communications, Control, and Computing (Allerton'05). Urbana-Champaign, IL. pp. 1–11.

[21] Sarkar, S., Chaporkar, P., and Kar, K. (2006) Fairness and throughput guarantees with maximal scheduling in wireless networks. Proc. 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt'06). Boston, MA. pp. 1–13.

[22] Chaporkar, P., Kar, K., and Sarkar, S. (2006) Achieving queue-length stability through maximal scheduling in wireless networks. Proc. Information Theory and Applications: Inaugural Workshop (ITA'06). San Diego, CA. pp. 1–4.

[23] Wu, X., Srikant, R., and Perkins, J. (2007) Scheduling efficiency of distributed greedy scheduling algorithms in wireless networks. IEEE Transactions on Mobile Computing. 6: 595–605.

[24] Gupta, A., Lin, X., and Srikant, R. (1997) Low-complexity distributed scheduling algorithms for wireless networks. Proc. Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'97). Kobe, Japan. pp. 1631–1639.

[25] Joo, C. (2008) A local greedy scheduling scheme with provable performance guarantee. Proc. ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC' 08). Hong Kong SAR, China. pp. 111–120.

[26] Bui, L., Eryilmaz, A., and Srikant, R. (2008) Asynchronous congestion control in multihop wireless networks with maximal matching-based scheduling. IEEE/ACM Transactions on Networking. 16: 826–839.

[27] Joo, C., Lin, X., and Shroff, N. B. (2009) Understanding the capacity region of the greedy maximal scheduling algorithm in multihop wireless networks. IEEE/ACM Transactions on Networking. 17: 1132–1145.

[28] Chen, L., Low, S. H., Chiang, M., and Doyle, J. C. (2006) Cross-layer congestion control, routing and scheduling in ad hoc wireless networks. Proc. Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'06). Barcelona, Spain. pp. 1–13.

[29] Penttinen, A., Koutsopoulos, I., and Tassiulas, L. (2006) Low-complexity distributed fair scheduling for wireless multi-hop networks. Proc.Workshop on Resource Allocation in Wireless Networks (RAWNET'06). Boston, MA. pp. 1–6.

[30] Lin, X. and Rasool, S. B. (2006) Constant-time distributed scheduling policies for ad hoc networks. Proc. IEEE Conference on Decision and Control (CDC'06). San Diego, CA. pp. 1258–1263.

[31] IEEE (2004), IEEE standard for local and metropolitan area networks - part 16: Air interface for fixed broadband wireless access systems. IEEE Std. 802.16. 857 p.

[32] Cao, M., Zhang, Q., and Wang, X. (2007) Analysis of ieee 802.16 mesh mode scheduler performance. IEEE Transactions on Wireless Communication. 6: 1455–1464.

[33] Wang, S.-Y., Lin, C.-C., Chu, H.-W., Hsu, T.-W., and Fang, K.-H. (2008) Improving the performance of distributed coordinated scheduling in ieee 802.16 mesh networks. IEEE Transactions on Vehicular Technology. 4: 2531–2547.

[34] Vejarano, G., Wang, D., and McNair, J. (2011) Stability region adaptation using transmission power control for transport capacity optimization in ieee 802.16 wireless mesh networks. Computer Networks (Elsevier). 55: 3694–3704.

[35] Vejarano, G. and McNair, J. (2007) An intelligent wireless mesh network backbone. Proc. 3rd International Conference on Wireless Internet (WICON'7). Austin, TX. pp. 1–5.

[36] Vejarano, G. and McNair, J. (2010) Queue-stability-based transmission power control in wireless multihop networks. Proc. Global Communications Conference, IEEE (GLOBECOM' 10). Miami, FL. pp. 1–5.

[37] Brzezinski, A., Zussman, G., and Modiano, E. (2006) Enabling distributed throughput maximization in wireless mesh networks: A partitioning approach. Proc. 12th Annual International Conference on Mobile Computing and Networking (MobiCom'06). Los Angeles, CA. pp. 26–37.

[38] Zussman, G., Brzezinski, A., and Modiano, E. (2008) Multihop local pooling for distributed throughput maximization in wireless networks. Proc. 27th Conference on Computer Communications IEEE (INFOCOM'08). Phoenix, AZ. pp. 1139–1147.

[39] Birand, B., Chudnovsky, M., Ries, B., Seymour, P., Zussman, G., and Zwols, Y. (2010) Analyzing the performance of greedy maximal scheduling via local pooling and graph theory. Proc. 29th Conference on Computer Communications IEEE (INFOCOM'10). San Diego, CA. pp. 26-37.

[40] Gao, Y., Zeng, Z., and Kumar, P. R. (2010) Joint Random Access and Power Selection for Maximal Throughput in Wireless Networks. Proc. 29th Conference on Computer Communications IEEE (INFOCOM'10). San Diego, CA. pp. 1-5.

[41] Monks, J. P., Bharghavan, V., and Hwu, W. (2001) A power controlled multiple access protocol for wireless packet networks. Proc. 20th Conference on Computer Communications. IEEE (INFOCOM'01). Anchorage, AK. pp. 219–228.

[42] Muqattash, A. and Krunz, M. (2003) Power controlled dual channel (pcdc) medium access protocol for wireless ad hoc networks. Proc. 22nd Conference on Computer Communications. IEEE (INFOCOM'03). San Francisco, CA. pp. 470–480.

[43] Muqattash, A. and Krunz, M. (2004) A single-channel solution for transmission power control in wireless ad hoc networks. Proc. 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'04). Roppongi, Japan. pp. 210–221.

[44] Ho, I.W.-H. and Liew, S. C. (2007) Impact of power control on performance of IEEE 802.11 wireless networks. IEEE Transactions on Mobile Computing. 6: 1245–1258.
[45] Jiang, L. B. and Liew, S. C. (2008) Improving throughput and fairness by reducing exposed and hidden nodes in 802.11 networks. IEEE Transactions on Mobile Computing. 7: 34–49.
[46] Wang, W., Srinivasan, V., and Chua, K.-C. (2008) Power control for distributed mac protocols in wireless ad hoc networks. IEEE Transactions on Mobile Computing. 7: 1169–1183.
[47] Behzad, A. and Rubin, I. (2007) Optimum integrated link scheduling and power control for multihop wireless networks. IEEE Transactions on Vehicular Technology. 56: 194–205.
[48] Karnik, A., Iyer, A., and Rosenberg, C. (2008) Throughput-optimal configuration of fixed wireless networks. IEEE/ACM Transactions on Networking. 16: 1161–1174.
[49] Vejarano, G. and McNair, J. (2010) WiMax-RBDS-Sim: An OPNET simulation framework for IEEE 802.16 mesh networks. Proc. 3rd International ICST Conference on Simulation Tools and Techniques (SIMUTools'10). Torremolinos, Malaga, Spain. pp. 1–10.
[50] Sahinidis Optimization Group (2012) Baron global optimization software. Available: http://archimedes.cheme.cmu.edu/baron/baron.html. Accessed 2012 March 23.