EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

An efficient admission control model based on dynamic link scheduling in wireless mesh networks

Juliette Dromard*, Lyes Khoukhi and Rida Khatoun

Abstract

Background: Wireless mesh networks (WMNs) are a very attractive new field of research. They are low cost, easily deployed, and a high-performance solution to last-mile broadband Internet access. In WMNs, admission control (AC) is one of the key traffic management mechanisms that should be deployed to provide quality of service (QoS) support for real-time traffic.

Results: In this paper, we introduce a novel admission control model, based on bandwidth and delay parameters, which integrates a dynamic link scheduling scheme. The proposed model is built on two different methods to access the medium: on a contention-based channel access method for control packets and on a dynamic time division multiple access (DTDMA) for data packets. Each time a new flow is admitted in the network, the WMN's link scheduling is modified according to the flows' requirement and network conditions while respecting the signal-to-interference-plus-noise ratio (SINR); this allows establishing collision-free transmissions.

Conclusions: Using extensive simulations, we demonstrate that our model achieves high resource utilization by improving throughput, establishing collision-free transmission, as well as respecting requirements of admitted flows in terms of delay and bandwidth.

Keywords: Wireless mesh networks; Admission control; Link scheduling

1 Introduction

Wireless mesh networks (WMNs) are autonomous networks, made up of mesh routers and mesh clients, where mesh routers have minimal mobility and form the backbone of WMNs. The bridge between the backbone mesh and other networks (e.g., Internet, cellular and sensor networks, etc.) is achieved through gateways. Mesh routers relay the data injected by mesh clients in a multihop *ad hoc* fashion until reaching a gateway. WMNs are low cost, easily deployed, self-configuring, and self-healing and enable ubiquitous wireless access. Indeed, they can extend Internet access in areas where cable installation is impossible or economically not sustainable such as hostile areas, battlefields, old buildings, rural areas, etc. [1].

However, due to a lack of centralized management, unfairness between flows created by the contention-based channel access, and unreliable wireless channels, the capacity of WMNs is limited. The capacity of a node tends to decrease with the number of hops which separates it from the gateway; this fact compromises the scalability of WMNs. Indeed, the throughput of each node decreases as $O(1/n)$, where n is the total number of nodes in the network [2]. That is why many applications with very strict constraints (e.g., VOIP and video streaming) cannot be deployed easily.

In order to solve the deployment issues of flows with very strict requirements in WMNs, several admission control (AC) schemes have been proposed in the literature [3-7]. AC schemes aim at guaranteeing flows' constraints in WMNs by accepting a new flow on the backbone mesh only if the latter is able to guarantee its quality of service (QoS) and the QoS of previously accepted flows. Note that most existing ACs in WMNs are based on a contention-based channel access method, carrier sense multiple access with collision avoidance (CSMA/CA) [6]. CSMA/CA was originally built for infrastructure wireless

*Correspondence: dromardj@utt.fr
University of Technology of Troyes, 12 rue Marie Curie, Troyes 10000, France

networks and turns out to be inappropriate in a multi-hop wireless network as it leads to low throughput and unfairness between nodes in WMNs [8].

To overcome the above limitations, several articles (e.g., [9-12]) propose to replace CSMA/CA by TDMA in WMNs. As TDMA is not a competition access scheme, it does not need methods to avoid collision (such as the backoff algorithm used in CSMA/CA) and can, thus, gain in throughput [12,13]. Furthermore, as it divides the access to the channel in time in order to avoid collisions, it also enables to limit packet loss rate [14]. However, in most existing TDMA schemes in WMNs, the applied link scheduling is fixed at the design stage of the network and does not evolve according to the traffic load; this may lead to network congestion. To alleviate these limitations, this paper proposes a novel admission control model based on dynamic link scheduling, which integrates bandwidth and delay as parameters. We note that a preliminary version of this work was published in [15]. Our model includes the following contributions:

- The use of the signal-interference-plus-noise-ratio (SINR) as the interference model in our AC. While most existing AC models (e.g., [4,5,16]) rely on either hop-based or distance-based interference model, our AC considers a more accurate interference model [17]: the SINR-based interference model. This allows establishing interference-free transmissions and increasing network throughput.
- An analytical formulation which allows computing the delay of any flow, knowing its scheduling over links it crosses. Integrated in an AC scheme, the latter accepts a flow only if its link scheduling respects the required delay. Furthermore, our analytical formulation can be integrated in the IEEE 802.11s mesh coordination function controlled channel access (MCCA) [18] in order to ensure that the flows' delay requirement is respected.
- A heuristic algorithm which allows dynamic link scheduling which respects traffic constraints in terms of delay and bandwidth. While most existing link scheduling solutions based on the SINR model do not consider dynamic traffic load in the network and propose fixed link scheduling, our algorithm updates link scheduling dynamically according to flows' requirements and traffic load; this prevents the network from congestion.

To the best of our knowledge, it is the first time that an AC scheme considers dynamic link scheduling based on the SINR interference model. Our solution allows to overcome the lack of throughput of existing ACs in WMNs as well as the lack of traffic adaptation according to the network's load and traffic types. The rest of the paper is organized as follows. In Section 2, we survey recent works related to link scheduling and admission control schemes and underline the necessity to integrate AC and link scheduling schemes into a unique solution. In Section 3, we present our system model and formulate our problem. In Section 4, we detail our proposed model. Section 5 evaluates the proposed admission control via simulations. Finally, Section 6 concludes this paper.

2 Related works

Several papers have addressed the problem of link scheduling to guarantee collision-free transmission. To deploy a link scheduling scheme in a WMN, the nodes must be synchronized and time must be divided into frames split into slots. Link scheduling schemes aim at selecting for each link in the network the slots in a frame during which the link is periodically activated while ensuring interference-free transmission and a maximum throughput in the network [14]. To avoid collisions, a link scheduling scheme should employ an interference model in order to establish which set of links can be activated simultaneously without causing any interference issue. The problem of link scheduling with the objective of maximizing the network throughput is known to be NP-hard, even with a simple interference model [12]. Thus, most existing works propose heuristic algorithms which produce close-to-optimal (sub-optimal) solutions. The efficiency of a sub-optimal algorithm is typically measured in terms of computational complexity (run time) and approximation factor (performance guarantee) [14]. Link scheduling schemes can be classified according to the interference model they are based on, which can be either a hop-based (e.g., [12,19]), a distance-based (e.g., [20,21]), or a SINR-based interference model (e.g., [10,22-24]).

In the hop interference model, a node can transmit successfully if no node, situated at k-hops or more from it, is activated at the same time [25]. For example, the request to send/clear to send (RTS/CTS) scheme respects the one-hop interference model. Indeed, to transmit data over a link, the sender and the receiver of the link must previously send RTS/CTS packets. Nodes situated at one hop of the sender or of the receiver which receives either a RTS or a CTS packet are then blocked to prevent interference during the link transmission. Figure 1 represents this phenomenon; it shows that if link A-B is active, the RTS/CTS scheme blocks the neighboring nodes (nodes in black) of node A and node B in order to avoid collisions.

In [10], the authors formulate the k-hop interference model as a k-valid matching problem in a network graph. They propose a scheduling scheme based on a greedy algorithm which computes sets of independent maximum k-valid matchings in the network graph. A maximum k-valid matching is the maximum set of edges which are at least k-hops from each other and which

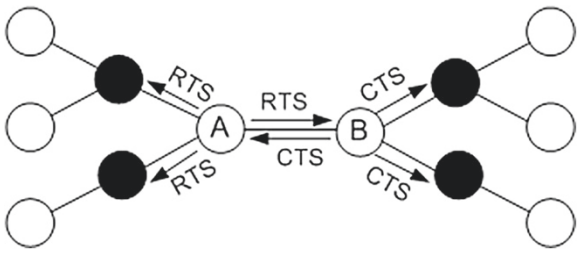

Figure 1 The RTS/CTS scheme ensures a one-hop interference model. Nodes in black are blocked and cannot accept or launch any transmission.

can be activated simultaneously during some slot(s). The algorithm searches for maximum k-valid matchings in order to optimize the network throughput. However, this solution can only be deployed to a limited number of topology.

The authors in [19] consider the problem of scheduling the links of a set of routes in a WMN while respecting the hop-based interference model and maximizing the network throughput. In their approach, an undirected graph G is built where each node represents a link to schedule; an edge may exist between two nodes if the links represented by these nodes interfere with each other when they are activated simultaneously. The authors show that the problem of scheduling the links of a set of routes can be considered as a problem of multicoloring the nodes of the graph G. They introduce two multicoloring-based heuristics in order to schedule the links of the WMN and study their performance. However, the scalability aspect is not respected in their approach because they only study WMNs made up of a few nodes.

In a distance-based model, a transmission is successful if the distance between a receiver and a transmitter is less or equal to the communication range R_c and if no other node is transmitting in the interference range R_i of the receiver; i.e., within a distance R_i from the receiver [17,20] (see Figure 2).

In [21], the authors propose new methods for computing upper and lower bounds on the optimal throughput for any given network and workload. They also introduce a conflict graph model based on the distance-based interference model to represent clearly interferences between links. In their proposed conflict graph $F(V, E)$, each node $v_i \in V$ of the conflict graph represents a direct link in the network. The model assumes that there exists an edge $e = (v_i, v_j)$ with $e \in E$ which joins up two links represented by nodes v_i and v_j, if these two links interfere with each other when they are activated simultaneously according to the distance-based interference model. The developed methods to compute upper and lower bounds on the optimal throughput assume that packet transmissions at the individual nodes can be finely

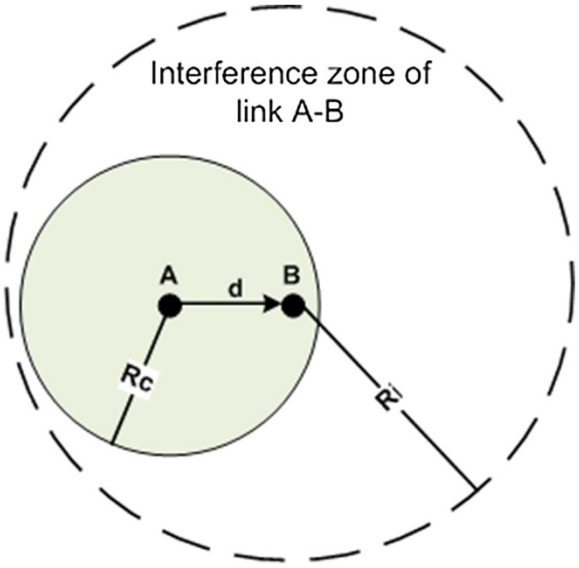

Figure 2 Communication range R_c and interference range R_i of a link A-B using the distance-based interference model.

controlled and carefully scheduled by an omniscient and omnipotent central entity, which is clearly unrealistic.

In [24], the authors investigate the problem of finding the link scheduling for a set of paths in a WMN relying on the distance based-interference model. They represent the issue by a mixed integer-nonlinear problem and propose heuristics based on Lagrangian decomposition to compute suboptimal solutions. They show that their solution is sub-optimal and can be rapidly computed in large WMNs. However, the interference model used in their solution is not the most realistic one and may lead to interference issues [26].

The SINR-based interference model assumes that a receiver successfully receives data if its SINR is greater than or equal to a certain threshold whose value can be given as physical layer properties of the network card [14]. The SINR-based model is not a local concept; indeed, any far away node can be involved in corrupting a transmission [23]. So the SINR-based model is less restrictive and more accurate than both the hop-based or distance-based models; however, it is more complex.

In [10], the authors present a centralized polynomial time algorithm for link scheduling using the SINR-based interference model. This algorithm schedules link by link; each link is scheduled at slots such that the resulting sets of scheduled transmission are feasible. To maximize the network throughput, this algorithm looks at minimizing *schedule length* (i.e., finding the shortest frame which enables to schedule every link). The authors formally prove, under uniform random node distribution, an approximation factor for the length of the schedule

relative to the shortest schedule possible under the SINR-based interference model. In their solution, the authors assume that flows' demands are known *a priori* by the scheduling module, which is an unrealistic assumption.

In [22], the authors study the limits of the distance-based interference model and propose a conflict-free link scheduling algorithm (CFLS) based on the Matroid theory. CFLS is a low conflict-free link scheduling algorithm with high spatial reuse. The authors argue that there is no known relation between schedule length and network throughput; so to maximize network throughput, they introduce a spatial reuse metric. Furthermore, they derive upper bounds on the running time complexity of their algorithm and prove that their CFLS algorithm can be solvable in polynomial time.

We note that while the first two interference approaches (i.e., hop-based and distance-based models) enable low computation, they can accept transmissions that lead to interference and may reject other transmissions that are interference free [17]. The SINR-based model is the most accurate model (even it is more complex). However, most existing works on link scheduling are static which means that the number of slots dedicated to each link does not evolve in time and with the network load. Thus, a link is dedicated the same number of slots when it is high loaded and low loaded which can lead, respectively, to congestion issues and bandwidth losses.

Admission control schemes aim at accepting a new flow in the network only if it can guarantee its delay and bandwidth and the delay and bandwidth of previously admitted flows. To decide whether a flow can be admitted along a given path, the admission control scheme must evaluate whether every node along the path has an available bandwidth sufficient to meet the new flow requirements. If it is the case, it accepts the new flow along this path; otherwise, it rejects it. The available bandwidth of a node can be defined as the maximum amount of bandwidth that a node can use for transmitting without depriving the reserved bandwidth of any existing flows [4] and so without causing any interference; thus, it depends mainly on the interference model considered. Furthermore, as AC schemes mainly differ from each other in their method of computing the available bandwidth of nodes [6] and so in the interference model they are based on, the choice of the interference model used to evaluate the bandwidth is of central importance in AC. In the AC models developed in [3-6], the authors reported that the available bandwidth of a node is mainly based on the channel idle time ratio (CITR). In the CITR-based scheme, the available bandwidth of a node is equal to the fraction of the idle time of its carrier sensing range multiplied by the capacity of its channel. Thus, CITR assumes that an interference occurs only when a node transmits simultaneously with another node situated in its carrier sensing range. So this scheme relies on the distance-based interference model. However, when a node senses its channel, it does not imply that it hears all nodes situated in its carrier sensing range as some nodes may be hidden. Thus, a node which applies CITR to compute its available bandwidth may not apply precisely the distance-based interference model due to the hidden node problem.

To overcome this issue, the authors in [3] propose a probabilistic approach to estimate the available bandwidth of a node which does not trigger any overhead. This approach is based on CITR and considers the impact of hidden terminals in WMNs. Upon this available bandwidth estimation, the authors design an admission control algorithm (ACA) which differentiates QoS levels for various traffic types.

In [5], the authors propose an admission control scheme which computes the available bandwidth of a node while considering its CITR and the spatial reuse issue. Indeed, as mentioned in [14,20], the distance-based interference model can be, in some situations, too cautious and can prevent some nodes from sending in parallel even though there is no risk of interference. Thus, to overcome this issue, the authors propose to compute the available bandwidth through passive monitoring of the channel and to improve the bandwidth estimation accuracy using a formula that considers possible spatial reuse from parallel transmissions. This solution can be integrated in networks with multirate nodes.

In CACP [4], the authors differentiate two types of bandwidth: the *available local available bandwidth* of a node based on the CITR which considers interference issue and the *available bandwidth* of a node which considers both blocking and interference issues. A blocking issue occurs when a node cannot continue to send a flow which has been previously admitted. To avoid this problem, the authors compute the available bandwidth of a node as the smallest available local bandwidth of all nodes situated in its carrier sensing range and itself.

In [27], the authors propose a fuzzy decision-based multicast routing resource admission control mechanism (FAST). In this solution, once every node on a flow path has accepted the flow in terms of bandwidth, the source has the final decision to accept or not the flow according to the decision made by the fuzzy decision scheme. This intelligent method selects among all the flows, which the source wants to send and which have been previously accepted by the nodes of their path, the optimal one in terms of delay, jitter, packet loss, and bandwidth. Once the optimal flow is chosen, the source can start sending it. The validation of this solution shows good results; however, the authors do not specify how the used values of the parameters (bandwidth, delay, jitter, packet loss) are obtained.

In [7], the authors propose an interference-aware admission control (IAC) for use in WMNs. The originality of their work lies in a dual threshold-based approach to share the bandwidth between neighbors; this sharing is essential to compute the available bandwidth of nodes. However, the IAC solution cannot deal with multirate nodes and does not consider the possibility of parallel transmissions which may lead to underestimation of the nodes' available bandwidth.

The AC schemes presented above, as most existing AC schemes, are based on CSMA/CA which is known to lead to poor throughput [10]. Indeed, CSMA/CA triggers interference and dedicates a huge amount of time to avoid collision (via backoff algorithm and RTS/CTS mechanism). To overcome these issues, the IEEE 802.11s standard [18] proposes the protocol MCCA which takes advantage of both admission control and link scheduling schemes in WMNs. In MCCA, nodes can reserve future slots in advance for their flows. To reserve a slot for a transmission, a node must first check if no node situated at two hops from it or from its receiver has already reserved the slot. Thus, MCCA is based on the two-hop interference model. However, MCCA may suffer from collisions due to hidden node problems [28] and does not specify any link scheduling algorithm [29]. In a previous work [15], we have proposed to integrate link scheduling in an admission control. However, in this previous work, the link scheduling scheme is totally distributed and integrates the distance-based interference model. A flow is admitted when there exists a path where every node is able to compute a link scheduling for this flow while respecting its requirements in terms of bandwidth. However, this solution generates an important overhead due to the broadcast of advertisement packets and lacks accuracy as it does not rely on a SINR-based interference model. Furthermore, it does not integrate the delay parameter in the admission control which can prevent the deployment of multimedia flows.

3 System model

In this section, we present our system model and formulate our link scheduling problem. The system model includes the interference model, the time division model, the link scheduling in terms of bandwidth, and the link scheduling in terms of delay. We also present and prove two theorems upon which our new method computes the delay of any flow knowing its scheduling. We then formulate our problem of admitting or rejecting a new flow along a certain path according whether there exists a scheduling over this path which respects the bandwidth and the delay of the new flow and of previously admitted flow, or not. In our work, we only consider the WMN's backhaul made up with mesh routers. Thus, the router to which a client is directly connected is considered as the sender of the flow. Every client wants to access the Internet, so no flow is directly exchanged between clients and every flow is sent via a gateway on the Internet. Mesh routers, also called nodes or stations in our paper, have all the same radio parameters.

3.1 Interference modeling

In our model, we consider that there is a directed link going from a node u_j to a node v_j if v_j receives successfully the data sent by u_j when u_j is the only transmitter node in the network. When a node is the only transmitter, its signal is supposed to be valid at a receiver if the signal-to-noise ratio (SNR) at the receiver is above a constant threshold β which depends on several parameters (e.g., data rate, modulation type, network card features). Let P be the power level of the nodes, V be the set of nodes in the network and u_j and v_j be two nodes (where $u_j,\ v_j \in V^2$). There is a directed link $e_i = (u_j,\ v_j)$ if [14]:

$$\frac{P}{d(u_j - v_j)^\alpha \times N} \geq \beta \tag{1}$$

where N is the ambient noise power level, α is the path loss exponent, and $d(u_j, y\, v_j)$ is the Euclidean distance between the two nodes. As all nodes have the same radio parameters, if the directed link (u_j, v_j) exists, then the directed link (v_j, u_j) exists too. Let E denote the set of links in the network such that:

$$E = \left\{ (u_j, v_j) \in V^2 | \frac{P}{d(u_j, v_j)^\alpha} \geq \beta N \right\} \tag{2}$$

According to the *physical interference model* [14,30], also called the SINR-based interference model, a set of links Γ can transmit simultaneously with success if every node u_j and node v_j such that $(u_j, v_j) \in \Gamma$ respects the following formula:

$$\frac{\frac{P}{d(u_j,v_j)^\alpha}}{N + \sum\limits_{(u_k,v_k)\in\Gamma-(u_j,v_j)} \frac{P}{d(u_k,v_j)^\alpha}} \geq \beta \tag{3}$$

However, we assume that a link transmission is successful when both data and acknowledgment packets are received successfully. So a transmission on a link (u_j, v_j) is successful if v_j receives the data sent by u_j and then u_j receives the acknowledgment sent by v_j. In our model, a transmission (i.e., data packet and its acknowledgment) on a link:

- Starts only at the beginning of a time unit (TU) interval (see Figure 3).
- Lasts the time of a TU (this will be detailed in the next sections, see Equation 6). Note that we assume that all data packets have the same size and thus have the same transmission time.

Hence, no transmitter u_j of links $(u_j, v_j) \in \Gamma$ can send in parallel with any receiver v_j of any links $(u_j, v_j) \in$

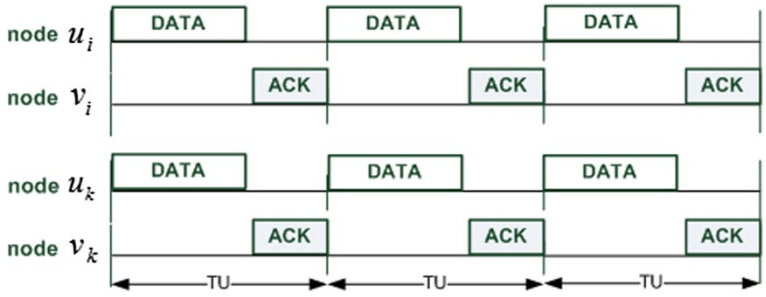

Figure 3 Links (u_i, v_i) and (u_k, v_k) transmit simultaneously. The data transmission does not overlap the ack transmission.

Γ (see Figure 3). From the above assumptions and the SINR-based model, we propose the following interference model: a transmission on a link (u_j, v_j) is successful if both packet data transmission (Equation 4) and acknowledgement transmission (Equation 5) are successful:

$$\frac{\frac{P}{d(u_j,v_j)^\alpha}}{N+\sum_{(u_k,v_k)\in\Gamma-(u_j,v_j)}\frac{P}{d(u_k,v_j)^\alpha}} \geq \beta \tag{4}$$

$$\frac{\frac{P}{d(v_j-u_j)^\alpha}}{N+\sum_{(u_k,v_k)\in\Gamma-(u_j,v_j)}\frac{P}{d(v_k,u_j)^\alpha}} \geq \beta \tag{5}$$

When a subset Γ of links are transmitting simultaneously, the transmission on the link $e_i = (u_j, v_j) \in \Gamma$ is successful if it respects Equations 4 and 5. Let I be the distance matrix of size $|V| \times |V|$. The value of each element i_{ij} of I is the Euclidean distance between node v_i and node v_j: when $i = j$, the value of the element i_{ij} is null, as the Euclidean distance between a node and itself is null.

3.2 Time division

In our model, we consider that the nodes of our network are synchronized and the time is split in two different intervals (see Figure 4):

- TU interval: a TU interval has a length equal to the time needed for the transmission of a packet. The formula to compute TU's length is presented hereafter in the paper.
- Transmission scheduling (TS) interval: a TS interval is composed of a fixed number N of TUs and is periodically repeated every $N \times TU$.

A TS interval is made up of two periods: the first one is the period of contention access to the channel denoted T_c and the second is the link scheduling period denoted T_s (see Figure 4). The T_c period is made up of N_c TUs (with $N_c < N$): during this period, nodes send packets using CSMA/CA, whereas the TS period is made up of N_s TUs (with $N = N_c + N_s$): during this period, nodes transmit data packets using TDMA.

Scheduling a link consists in selecting the TUs in the T_s interval during which the link will be activated without any risk of interference. In order to give more flexibility in the selection of TUs, the length of a TU must be as short as possible; a TU is equal to the time of a packet transmission, which is the same for all packets:

$$TU = T_{\text{packet}} = T_{\text{difs}} + \frac{L+H}{C} + T_{\text{ack}} + T_{\text{sifs}} + 2\times T_{\text{PLCP}} \tag{6}$$

where L and H, respectively, are the size of a data packet and its header; T_{difs} and T_{sifs}, respectively, are the interframe space time of DIFS and SIFS which are defined in the IEEE 802.11 standard; T_{ack} is the transmission time of an acknowledgment; and T_{plcp} is the transmission time of the physical layer convergence protocol (PLCP) header [4]. The requirements of a flow f are expressed by the double $\triangle_f = (br_f, dr_f)$, where br_f represents the minimum bandwidth required by f and dr_f represents the maximum delay required. The set of flows which are in process are

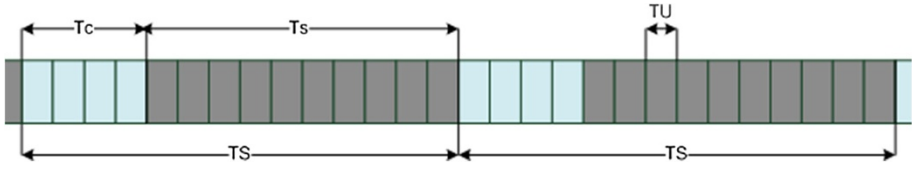

Figure 4 Time split in TS periods made up of N TUs with $N = 14$. A T_c interval is made up of N_c TUs with $N_c = 4$, and a T_s interval is made up of N_s TUs with $N_s = 10$.

denoted F. In the sequel, we present and prove two theorems. These theorems are at the base of our method to compute the delay of any flow knowing its scheduling.

3.3 Link scheduling in terms of bandwidth

For each flow, every node u_i along its path (except the destination) reserves a fixed number of TUs (denoted TU_f) in the TS interval in order to respect the flow requirements in terms of bandwidth. For a flow f with a rate br_f, TU_f can be computed with the following formula:

$$TU_f = \left\lceil \frac{br_f}{L} \times TS \right\rceil \tag{7}$$

where $\lceil\rceil$ is the ceiling function and TS the length (in seconds) of a TS interval. The scheduling of a flow f over a link e_i is denoted by a $z-$ tuple, $r_f(e_i) = (tu(e_i)_1, \ldots, tu(e_i)_z)$ where z is equal to TU_f. Each element $tu(e_i)_j$ represents the position of a TU in the TS interval during which node u_i such that $e_i = (u_i, v_i)$ must send a packet of flow f if it possesses any in its queue. These elements are ordered in ascending order such that:

$$\forall j \in [\,1,\ TU_f - 1\,] \text{ and } \forall e_i \in l_f,\ tu(e_i)^f_j < tu(e_i)^f_{j+1} \tag{8}$$

For example, in Figure 5, the schedule of flow f over link e_0 is $r_f(e_0) = (2,\ 6,\ 8)$.

3.4 Link scheduling in terms of delay

The link scheduling must respect the delay dr_f that every flow $f \in F$ requires. The delay of a flow depends of its scheduling over every link along its path. A path of a flow f is denoted as $l_f = (e_0, e_1, \ldots, e_n)$ and must verify the following constraints:

$$\begin{cases} \forall e_i \in l_f, & e_i \in E \\ \forall i \in [\,1, n-1\,] \ et\ e_i = (u_i, v_i), & e_{i+1} = (v_i, v_{i+1}) \\ \forall (i,j) \in [\,1, n]^2 \ et\ i \neq j, & e_i \neq e_j \\ e_n = (u_n, v_n), & v_n \in V^* \end{cases} \tag{9}$$

where V^* is the set of gateways in the network. In the following, the transmitter node u_i and the received node v_i of a link e_i possess the same index i as that of the link it belongs to. Furthermore, this index represents the link position on the path of a flow f; the index starts at 0. We denote p^f_j the jth packet of a flow f. We define the three following delays:

- *Delay of a packet at a link:* The delay of a packet p^f_j at a link e_i (with $e_i = (u_i,\ v_i)$) represents the time between the packet's arrival at the link's start node u_i and its correct delivery to the link's end node v_i and is denoted by $d(e_i, p^f_j)$.
- *Delay of a packet:* It is the time that takes a packet p^f_j to cross all links along its path. It is denoted by $d(p^f_j)$ and can be computed as follows:

$$d(p^f_j) = \sum_{\forall e_i \in l_f} d(e_i, p^f_j) \tag{10}$$

- *Delay of a flow:* The delay of a flow f is the maximum delay taken by a packet of this flow; it is denoted by $d(f)$. So if a flow f sends n packets, the delay of this flow can be computed as follows:

$$d(f) = \max\{S\} \text{ with } S = \{\forall j \in \mathbb{N} \text{ and } \forall j \in [\,0, n-1\,] |\ d(p^f_j)\} \tag{11}$$

The notations used in this section are presented in Table 1.

In the sequel, we fix the starting time (i.e., $t = 0$) at the beginning of the TS at which the first packet of flow f is sent over the link e_0. To compute the delay of any flow $f \in F$, we need to introduce the following assumptions:

- The transmitter node u_0 of link e_0 sends a packet of flow f at each reserved TU for f.
- Node u_0 receives a packet of f just before every TU it has reserved for f. Thus, the delay of every packet of f at edge e_0 is equal to one TU.
- u_0 sends its first packet at the first TU reserved for f, i.e., at the $(tu(e_0)_1)$th TU of the first TS interval, $tu(e_0)_1 \in r_f(e_0)$.
- Every node u_i has a FIFO queue whose length is equal $q(e_i, t)$ at time t. The length of every node's queue is initialized at 0, i.e., $q(e_i, 0) = 0, \forall i \in \mathbb{N}$.

Let t_j be the time such that $t_j = \mathrm{TS} \times j\,, \forall j \in \mathbb{N}$. Thus, the time t_j is the beginning of the $(j+1)$th TS interval. In the following, we introduce two theorems. These theorems are at the base of our method to compute the delay of any flow knowing its scheduling. The first theorem asserts that when a flow enters the network, it becomes stable at a link only after a certain time; once a flow is stable at a link, the start node of the link sends a packet of the flow at every reserved TU and has a queue's length at the

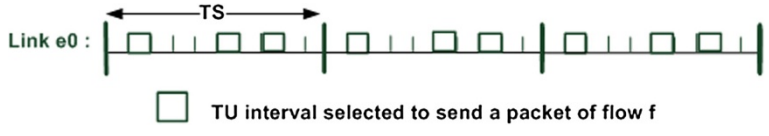

Figure 5 Link e_0 scheduling for flow f.

Table 1 Notations

Notation	Description
$r_f(e_i)$	Represents the set of the reserved TU_f TUs during which link e_i has to send packets of flow f
$tu(e_i)_j$	Position of the jth TU in the TS interval reserved by link e_i
p_j^f	The jth packet of the flow f
$d(e_i, p_j^f)$	The delay at the link e_i of the (p_j^f)th packet sent by flow f
$d(p_j^f)$	Delay of the packet p_j^f
t_j	Time t such that $t = \text{TS} \times j$
$d(e_i)_j$	The delay at link e_i of a packet sent by e_i at the $(tu(e_i)_j)$th TU of a TS interval during the stable period of e_i
$q(e_i, t)$	Length of the node u_i's queue at time t

beginning of every TS of the same size. The first theorem enables to prove the second theorem. The second theorem asserts that once a flow is stable at a link, the delay of its packets at this link is periodic, of period one TS, i.e., every packet sent at the same reserved TU of any TS interval gets the same delay at this link. Thus, according to the second theorem, the delay of a packet at a link takes only TU_f different values. By computing these TU_f values for every link, we get the delay of every packet at every link. By adding up the delay a packet gets at every link, we can get the delay of the packet. Then, the delay of the flow is obtained by extracting the highest delay among the delay of every packet of the flow.

3.4.1 Transition and stable periods

Theorem 1. *Before time t_i, every link $e_i(\forall e_i \in l_f)$ is in a transition period, i.e., its start node u_i possesses at the beginning of each TS interval a queue whose length is less or equal to $q(e_i, t_i)$ (queue's length of u_i at t_i) and does not send a packet at each of its reserved TU. From t_i, every link $e_i(\forall e_i \in l_f)$ is in a stable period, i.e., it sends a packet at each of its reserved TU and the length of its queue at the beginning of every TS interval is fixed and equal to $q(e_i, t_i)$.*

We can observe from Figure 6 the transition period of link e_1 which lasts until the end of the first TS interval and the one of link e_2 which lasts until the second interval. During link e_2's transition period, we can observe that its start node does not send a packet at each of its reserved TU. Furthermore, during its stable period, it sends a packet at each reserved TU and possesses at the beginning of every TS interval a queue of length 1.

Proof. Let us prove Theorem 1 by recurrence. First, Theorem 1 is valid for link e_0 as we have previously assumed that u_0 sends a packet at each of its reserved TU and the delay of any packet of flow f is one TU. Thus, from t_0, node u_0 sends a packet at every reserved TU and has at the beginning of each TS interval a queue of length 0. □

Now, let us prove Theorem 1 for every link $e_i \in l_f - \{e_0\}$. At t_{i-1}, u_i possesses a queue of length $q(e_i, t_{i-1})$. This queue length is inferior or equal to the one it would have obtained if u_{i-1} has sent a packet at each TU it has reserved since t_0. Let us assume that u_{i-1} has sent a packet at each reserved TU since t_0. During the first TS interval, u_i receives TU_f packets and forwards z. This implies that there exist a subset of $r_f(e_i)$ that we denote $r_f^1(e_{i-1}) = (tu(e_{i-1})_1^1, \ldots, tu(e_{i-1})_z^1)$ and a subset of $r_f(e_i)$ denoted $r_f^1(e_i) = (tu(e_i)_1^1, \ldots, tu(e_i)_z^1)$ such that:

$$\forall j \in [1, z],\ tu(e_{i-1})_j^1 < tu(e_i)_j^1 \tag{12}$$

Thus, when u_i receives a packet at the $(tu(e_{i-1})_j^1)$th TU, it can forward it at the $(tu(e_i)_j^1)$th TU as this TU is situated after. Thus, at t_1, u_i possesses a queue of length $TU_f - z$. During the second TS interval, u_i first sends the $TU_f - z$ packets it possesses in its queue at the beginning of the TSl. Then, it has still z available TUs denoted $r_f^2(e_i) = (tu(e_i)_1^2, \ldots, tu(e_i)_z^2)$ and which is a subset of $r_f(e_i)$. As it is the z last reserved TU of u_i, we get:

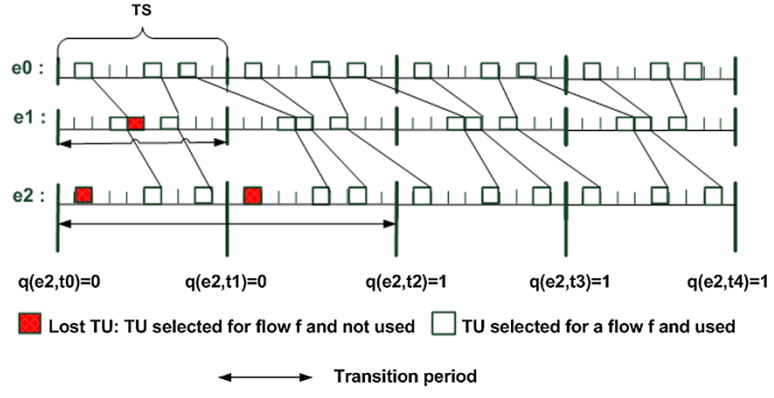

Figure 6 Packet transmission of flow f whose path flow is $l_f = (e_0, e_1, e_2)$.

$$\forall j \in [1, z],\ tu(e_{i-1})_j^1 < tu(e_i)_j^1 < tu(e_i)_j^2 \quad (13)$$

Thus, u_i can send a packet at each of its z available TUs. During this second TS, u_i sends a packet at each of its reserved TU and possesses at the end of the TS in its queue $TU_f - z$ packet. As the queue length of u_i is the same at t_2 and t_1, it implies that from t_1, u_i possesses at the beginning of each TS interval $TU_f - z$ packets in its queue. We can thus conclude that if u_{i-1} had sent a packet at each of its reserved TU since t_0, u_i would have had a queue of length $TU_f - z$ at t_{i-1}. Thus, at t_{i-1}, u_i possesses a queue of length $q(e_i, t_{i-1}) \leq TU_f - z$. From t_{i-1} to t_i and according to Theorem 1, u_{i-1} sends a packet at each of its reserved TU. During this TS, u_i first sends the $q(e_i, t_{i-1})$ packets it has in its queue at the beginning of the TS. Thus, it has still $TU_f - q(e_i, t_{i-1})$ TUs available in the current TS interval which can be represented by a subset of $r_f(e_i)$ denoted $r_f^3(e_i) = (tu(e_i)_1^3, \ldots, tu(e_i)_{TU_f - q(e_i,t_{i-1})}^3)$. As these TUs are the $TU_f - q(e_i, t_{i-1})$ last reserved TUs of u_i and that $TU_f - q(e_i, t_{i-1}) \leq z$, we get $r_f^3(e_i) \subset r_f^2(e_i)$; then according to Equation 13, u_i can send z packets among the TU_f it has received during the TS. Thus, from t_{i-1} to t_i, u_i does not send a packet at each reserved TU and link e_i is still in transition. At t_i, u_i possesses $TU_f - z$ packets in its queue, as we have previously seen; once it gets $TU_f - z$ TUs at the beginning of a TS interval, u_i then sends a packet at every reserved TU and possesses a queue of length $TU_f - z$ at the beginning of every TS interval. From t_i, link e_i gets stable. Thus, we have proved Theorem 1 by recurrence as it is true for link e_0 and for any link $e_i \in l_f - \{e_0\}$ if its previous link e_{i-1} respects the theorem.

3.4.2 Periodicity of packet delay

Theorem 2. *When a link e_i becomes stable, the delay of packets at this link gets periodic, i.e., every packet of flow f that e_i's start node, u_i, sends at the reserved TU $tu(e_i)_j (j \in [1, TU_f])$ of any TS has the same delay at the link e_i (denoted $d(e_i)_j$).*

Proof. From Theorem 1, we know that every link $e_i = (u_i, v_i)$ after time t_i is activated at each of its reserved TU and that u_i has a queue of length $q(e_i, t_i)$ at the beginning of each TS interval. Thus, after time t_i, when node u_{i-1} sends a packet at the $(tu(e_{i-1})_j)$th TU of a TS interval, node u_i forwards it at the $(tu(e_i)_{j'})$th TU such that $j' = (j + q(e_i, t_i))\%TU_f$:

- Of the current TS interval if $tu(e_{i-1})_j < tu(e_i)_{j'}$
- Of the next TS interval if $tu(e_{i-1})_j > tu(e_i)_{j'}$

□

For every link $e_i \in \{l_f\}$, we denote $\varphi_{e_i} : r_f(e_{i-1}) \to r_f(e_i)$ a bijective function which associates each position of a TU reserved by link e_{i-1} with one reserved by link e_i such that when e_{i-1} sends a packet at the $(tu(e_{i-1})_j)$th TU of a TS interval situated after t_i, link e_i forwards it at the $(tu(e_i)_{j'})$th TU of the current or of the next TS interval with $tu(e_i)_{j'} = \varphi_{e_i}(tu(e_{i-1})_j), \forall j, j' \in [1, TU_f]^2$. The inverse function of $\varphi_{e_i} : r_f(e_{i-1}) \to r_f(e_i)$ is $\varphi_{e_i}^{-1} : r_f(e_i) \to r_f(e_{i-1})$. During the stable period of a link, the delay $d(e_i)_j$ of a packet e_i sent by its start node u_i can be computed via the following equation:

$$d(e_i)_j = \begin{cases} (tu(e_i)_j - \varphi_{e_i}^{-1}(tu(e_i)_j) \times TU \\ \quad \text{if } tu(e_i)_j > \varphi_{e_i}^{-1}(tu(e_i)_j) \\ (TS + (tu(e_i)_j - \varphi_{e_i}^{-1}(tu(e_i)_j)) \times TU \\ \quad \text{if } tu(e_i)_j < \varphi_{e_i}^{-1}(tu(e_i)_j) \end{cases} \quad (14)$$

Recall that Theorem 1 asserts that the queue length of a node u_i at the beginning of a TS interval in the transition period is less than (or equal to) that in the stable period. It implies that when u_i receives a packet during its transition period, this packet waits less (or the same) time before being sent than during the stable period; the delay for this packet is at link e_i less or equal to the delay in the stable period. Thus, if we denote p_y a packet that u_i sends during a stable period at the jth TU and p_z a packet that u_i sends during a stable period at the jth TU with $j \in [1, nb(TU)_f]$, then:

$$d(e_i, p_z)) \leq d(e_i, p_y) \text{ and } d(e_i, p_y) = d(e_i)_j \quad (15)$$

Equations 14 and 15 prove Theorem 2.

3.4.3 Packet and flow delay

Recall that the delay of a packet is the sum of the packet delay at each link it crosses. Let p_y be the yth packet sent by u_0 such that $y = z \times TU_f + j$ with $j \in [1, TU_f]$ and $z \geq n$; the delay of this packet is:

$$d(p_y) = TU + d(e_1)_{j^1} + d(e_2)_{j^2} + \cdots + d(e_n)_{j^n} \quad (16)$$

where j^1 is the index of the reserved TU at which e_1 sends the packet p_y (i.e., $tu(e_1)_{j^1} = \rho_{e_1}(tu(e_0)_j)$), j^2 is the index of the reserved TU at which e_2 sends the packet p_y (i.e., $tu(e_2)_{j^2} = \rho_{e_1}(tu(e_1)_{j^1})$), etc. Thus, from t_n, there is TU_f different delays for a packet depending at which TU in the TS interval u_0 sends it. The delay of a packet sent at the $(tu(e_0)_j)$th TU of any TS interval ($\forall j \in [1, TU_f]$) in the transition period is less than or equal to that in the stable period. Indeed, at an intermediate node, every packet has less time to wait as the queue length at the beginning of each TS interval in the transition period is less than or equal to that in the stable period, as illustrated in Equation 15. Recall that we define the delay of a flow as the maximum delay that can reach a packet of the flow; thus,

the delay of a flow can be computed with the following equation:

$$d_f = \max(p_j^f) \text{ with } j \in [1, n] \text{ and } n \text{ the total number of packets of } f \quad (17)$$

3.5 Link scheduling

Let S be the scheduling matrix of size $|E| \times N$ which represents the scheduling made for every flow $f \in F$. The value of each element s_{ij} of S indicates whether the ith link e_i has scheduled a transmission at the jth TU of the TS interval:

$$s_{ij} = \begin{cases} f & \text{if } tu_j \in r_f(e_i) \\ 0 & \text{otherwise} \end{cases} \quad (18)$$

where tu_j represents the jth TU of a TS interval. The matrix S must satisfy the following constraints:

- The conflict-free constraint: Every link must check the SINR-based model, and so inequalities 4 and 5 and can only be scheduled during TUs of the link scheduling period T_s.
- The bandwidth constraint: For every flow $f \in F$, every link $e_i \in l_f$ (with l_f respecting the constraints expressed in 9) must reserve TU_f slots for f.
- The flow delay constraint: Every flow $f \in F$ must possess a delay inferior or equal to its requirements.

Each time a new flow f is admitted on the network, the scheduling matrix S^{old} must be updated such that all the scheduling made for this flow over every link $e_i \in l_f$ are included to form the new schedule matrix S^{new}. The elements of S^{new} are:

$$s_{ij}^{\text{new}} = \begin{cases} s_{ij}^{\text{old}} & \text{if } e_i \notin l_f \text{ or } j \notin r_f(e_i) \\ f & \text{if } e_i \in l_f \text{ and } j \in r_f(e_i) \end{cases} \quad (19)$$

At the end of the transmission of f, the schedule S^{old} is updated such that all the scheduling made for f over every link $e_i \in l_f$ are deleted to form the new scheduling matrix S^{new}.

3.6 Problem formulation of the admission control of a flow

As we have explained in the previous section, the requirements of a flow f are expressed by $\triangle_f = (br_f, dr_f,)$. We therefore propose a method which aims at finding for every requirement $\triangle_f$ of a new flow f and a path l_f the scheduling for every link on the flow path, i.e., to find $r_f(e_i)$, $\forall e_i \in l_f$ while respecting the three following constraints:

1. $|r_f(e_i)| = TU_f$.
2. $d_f \leq dr_f$.
3. S^{new} is feasible.

For a new flow, if every link on its path can find a schedule for this flow which satisfies these constraints, then the flow is admitted on the network; otherwise, the flow is rejected. Constraint 1 enforces that every link along the path respects the link requirements in terms of bandwidth. Constraint 2 enforces that the delay of the flow computed via Equation 17 respects the delay required by the new flow. Constraint 3 enforces that the new scheduling matrix of the network computed via Equation 19 is feasible, i.e., the scheduling made for previously accepted flows and the new flow are interference free.

4 Admission control

Our proposed admission control scheme is based on the reactive routing protocol AODV; however, packets have been modified in order to meet our approach requirements. The admission control takes place in three steps:

- Route discovery
- Link scheduling
- Route selection

4.1 Route discovery

During this phase, a source node broadcasts a route request (RREQ) packet which contains the source sequence number (to uniquely identify each packet), the time-to-live (TTL), the bandwidth, and the delay required by the new flow. Each node which receives the RREQ adds its identification *id* and rebroadcasts it if:

- The TTL has not expired.
- The node is not the gateway.
- It is the first time the node receives the RREQ.
- It succeeds the partial admission control.

The partial admission control aims at filtering flows' requests which could not be in the sequel accepted. Every node which receives a RREQ for a flow carries out a partial admission control on the flow; it checks whether the number of available TUs per T_s is superior or equal to the number TU_f of TUs required by the flow. The number TU_{av} of available TUs for a node in a T_s is equal to the number of TUs during which the node is neither a transmitter nor a receiver. If $TU_{av} \geq TU_f$, then the node has enough available bandwidth to satisfy the flow requirements and thus the partial admission control is achieved; otherwise, the node has not enough available bandwidth and thus the RREQ is dropped.

4.2 Link scheduling

When a RREQ reaches a gateway access point, the latter starts the admission control and the link scheduling process. Note that gateways are the only nodes which possess the I matrix which enables checking whether a set

of links can transmit simultaneously without interference. As link scheduling is computed at gateways and as the latter need the scheduling matrix S up to date to compute valid new link scheduling, thus, gateways have to be kept informed (for instance, via notification exchanges) about when a new flow is scheduled or stopped so that all gateways in the network have the same updated scheduling matrix. Recall that the link scheduling and the admission control are realized simultaneously. Indeed, a flow f is admitted if scheduling for every link on the flow path (i.e., $\forall e_i \in l_f$, $r_f(e_i)$ exists and satisfies the constraints cited in Section 3.3). To find a feasible link scheduling, we propose a simple greedy algorithm (Algorithm 1).

Algorithm 1 Admission control algorithm based on link scheduling

Require: $C, TU, TS, P, \beta, \alpha, N, dr_f, br_f, l_f$, and the matrix S, I

Ensure: S^{new}

1: Compute TU_f
2: Compute for every link e_i its linear matrix $Av(e_i)$ which specifies whether the jth TU of e_i's TS interval is available or not. If a link possesses no available TU, return null.
3: The reservations at every link $e_i \in l_f$ are initiated at empty, i.e., $r_f(e_i) = \varnothing$.
4: **for** $r = 1, r \leq TU_f, r++$ **do**
5: Update $Av(e_0)$ and check if has still available TU(s). If it has not, return null.
6: Pick randomly one available TU of $Av(e_0)$: $av(e_0)_j$.
7: Check if e_0 can be activated during the $(av(e_0)_j)$th TU according to the SINR-based interference model. If it is the case, e_0 reserves this TU, i.e., $r_f(e_0) = av(e_0)_j$; otherwise, return to line 5.
8: **for** $i = 1, i \leq n, i++$ **do**
9: Update $Av(e_i)$ and check if has still available TU(s). If it has not, return null.
10: Find $av(e_i)_j$ the available TU in $Av(e_i)$ which is the closest TU to the one that has just reserved e_{i-1} and which is situated after it. If there does not exist any, find $av(e_i)_j$ in $Av(e_i)$ such that it is the first available TU for e_i.
11: Check whether link e_i can be activated during the $(av(e_i)_j)$th TU such that the resulting S^{new} is feasible; if it is the case, link e_i reserves it, i.e., $tu(e_i)_r = av(e_i)_j$; otherwise, return to line 8
12: **end for**
13: **end for**
14: Compute the delay d_f of the flow from the source to the link with Equations 14, 16, and 17. If $d_f > dr_f$, then return null.
15: Update the scheduling matrix and return S^{new}

The algorithm returns null if the flow is rejected; otherwise, it returns a new scheduling matrix (which integrates the link scheduling made for the flow). The algorithm can be divided in three major steps. In a first step, we initiate the parameters used in the algorithm. We compute TU_f, the number of TUs that each link must reserve to respect the flow constraints in terms of bandwidth (see Equation 7). We then compute for every link $e_i \in l_f$ its linear matrix of available TUs $Av(e_i)$; each element $av(e_i)_j$ indicates whether the jth TU of the TS interval of link e_i is available ($av(e_i)_j = 0$) or not ($av(e_i)_j = 1$). An available TU for a link $e_i = (u_i, v_i)$ is a TU which has not been reserved by any link possessing either u_i or v_i or both as a receiver or a transmitter. In a second step, we reserve the TU_f TUs required by the flow on every link on the flow path. First, an available TU of link e_0 (i.e., the first link on the flow path) is randomly selected via its linear matrix of available TUs $Av(e_0)$. Then, the algorithm checks if e_0 can be activated during this selected TU (i.e., if the resulting scheduling matrix is feasible). If it is the case, the algorithm reserves for e_0 this TU; otherwise, it chooses randomly another possible available TU till e_0 has no more available TU in $Av(e_0)$ that it has not tested. Once an available TU has been reserved for e_0, we select one available TU for each upcoming link on the path, such that the selected TU is the closest to the TU that the previous edge has just chosen. The idea is to get the shortest feasible delay for a packet at each link. The algorithm then checks whether the link can be activated during this TU (i.e., if the resulting scheduling matrix is feasible). If it is the case, then the algorithm goes on; otherwise, it chooses another possible available TU till e_i has no more available TU. Each link on the flow path is processed till every link has reserved one TU. The algorithm uses the same method to reserve the other $TU_f - 1$ TUs required by the flow at every link along its path. If it succeeds in scheduling every link, then the algorithm enters its third step. In a third step, the algorithm checks whether the link scheduling that has just been realized respects the flow's requirements in terms of delay. First, it computes the start node queue length of every link during its stable period. Then, it can compute for every link $e_i \in l_f - \{e_n\}$ its function $\varphi_{e_i} : r_f(e_{i-1}) \rightarrow r_f(e_i)$. Thanks to it and via Equation 14, it gets the delay of every packet at a link. Then, it computes the TU_f possible delays of any packet in a stable period, thanks to Equation 16. Finally, by applying Equation 11, it gets the delay of the flow. If the delay of the flow is superior to the required delay, then the algorithm returns null; otherwise, it returns the new scheduling matrix and the flow is accepted. This algorithm can be solved in a polynomial time.

For better understanding of our admission control algorithm, let us consider an example. The values of the different parameters of this example are chosen in order

to simplify the algorithm. At the beginning, we are in the following situation (see Figure 7): the set of nodes of the WMN is $V = u_0, u_1, u_2, u_3$ and the set of links is $E = \{(e_0 = (u_0, u_1), e_1 = (u_1, u_0), e_{2=}(u_1, u_2), e_3 = (u_2, u_1), e_4 = (u_2, u_3), e_5 = (u_3, u_2)\}$, a flow f_1 has already been admitted, and the reservations made for this flow are represented in Figure 7. We want to admit a new flow f_2, which required delay $dr_{f_2} = 150$ ms and required bandwidth $ibr_{f_2} = 100$ kbit/s along a path $l_{f_2} = (e_0 = (u_0, u_1), e_2 = (u_1, u_2), e_4 = (u_2, u_3))$. The algorithm takes in input $C = 1$ Mbit/s, $TU = 1$ ms, $TS = 10$ ms, $P = 15$ dBm, $\beta = 20$, $\alpha = 2$, $N = -90$ dBm, dr_{f_2}, br_{f_2}, l_{f_2}, the distance matrix I, and the scheduling matrix S. The distance matrix is as follows:

$$I = \begin{matrix} 0 & 100 & 200 & 300 \\ 100 & 0 & 100 & 200 \\ 200 & 100 & 0 & 100 \\ 300 & 200 & 100 & 0 \end{matrix}$$

where each element $i_{ij} = d(u_{i-1}, u_{j-1})$. The scheduling matrix is as follows:

$$S = \begin{matrix} Nc & Nc & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Nc & Nc & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Nc & Nc & 0 & 0 & 0 & 0 & 0 & 0 & f_1 & f_1 \\ Nc & Nc & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Nc & Nc & 0 & 0 & 0 & 0 & f_1 & f_1 & 0 & 0 \\ Nc & Nc & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

where each element s_{ij} represents the jth TU of the TS interval of the link e_{i-1}. If $s_{ij} = Nc$, the jth TU of the TS interval is dedicated to control packets. If $s_{ij} = 0$, the link e_{i-1} does not send any packet at the jth TU of the TS interval. If $s_{ij} = f_z$, the link e_{i-1} sends a packet of flow f_z during the jth TU of the TS interval. In a first step, the algorithm computes the number of TUs required by this flow per TS using Equation 7, $TU_{f_2} = 1$. Then, the linear matrix of available TUs of every link on the flow path is computed: $Av(e_0) = (1, 1, 0, 0, 0, 0, 0, 0, 1, 1)$, $Av(e_2) = (1, 1, 0, 0, 0, 0, 1, 1, 1, 1)$, and $Av(e_4) = (1, 1, 0, 0, 0, 0, 1, 1, 1, 1)$. The reservations of every link $e_i \in l_{f_2}$ are initiated at empty, i.e., $r_{f_2}(e_i) = \varnothing$.

In a second step, the TU_{f_2} TUs required by the flow on each link $e_i \in l_{f_2}$ must be reserved. First, the algorithm checks if there is available TU(s) in the linear matrix $Av(e_0)$. As it is the case, an available TU of the link e_0 is randomly picked up, for example, the 7th TU. Then, the algorithm checks if e_0 can be activated during the 7th TU by verifying if the inequalities 4 and 5 are respected. As the inequalities are not respected, another available TU is so randomly picked up, for example, $TU = 5$. This time, the inequalities 4 and 5 are respected, and the TU is reserved, $r_{f_2}(e_1) = (5)$. Then, the matrix of available TUs of link e_2 is updated while considering the reservations that we have previously made; thus, $Av(e_2) = (1, 1, 0, 0, 1, 0, 1, 1, 1, 1)$. The available TU of link e_2 which is the closest to the one reserved by link e_0 and situated just after it is picked up; it is the 6th TU. The activation of this TU by link e_2 respects the inequalities 4 and 5, and this TU is reserved; $r_{f_2}(e_2) = (6)$. Link e_4's linear matrix is updated while considering the reservations that have been previously made; thus, $Av(e_4) = (1, 1, 0, 0, 1, 1, 1, 1, 1, 1)$. As there exists no available TU of link e_4 situated after the TU that has just chosen e_2 (the 6th TU), the first available TU of e_4 is picked up, which is the third TU of its TS interval. The inequalities 4 and 5 are respected, and the third TU is reserved; $r_{f_2}(e_4) = (3)$. If the flow had required more TU per TS than one, we would have returned to the second step in order to reserve for each link another TU. In a third step, we must check if the reservations that have just been made for flow f_2 respect its required delay. As we have previously made the assumption that the start node of link e_0 receives a packet just before each of its reserved TUs, the delay of every packet at e_2 is one TU, $d(e_2)_1 = 1\ TU$. We must now compute the delay of every packet of f_2 at every link on the flow path once every link is in their stable period. This computation is simplified as every link has reserved one TU per TS; thus, every packet

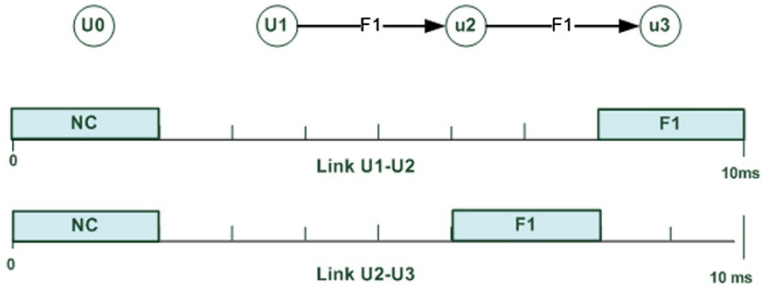

Figure 7 The TS intervals of the two links over which the flow f_2 is sent.

Figure 8 Chain topology.

at a link gets the same delay. Link e_2 has a queue at the beginning of each TS interval of length 0 as its reserved TU is situated just after that of e_0; thus, $\varphi_{e_2}(5) = 6$ and so according to Equation 14, the delay of the packet at the link e_2, $d(e_2)_1 = 1\ TU$. Link e_4 has, at the beginning of each TS interval, one packet in its queue. Indeed, when it receives a packet, it must wait for the next TS interval to send it; thus, $\varphi_{e_4}(6) = 3$ and according to Equation 16, the packet's delay at link e_4 is $d(e_4)_1 = 7\ TU$. Thus, the delay of any packet p_z of flow f_2 during the stable period is $d(p_z) = d(e_1)_1 + d(e_2)_1 + d(e_4)_1 = 1 + 1 + 7 = 9\ TU$. As only one packet is sent by TS, according to Equation 17, $d(f_2) = 9$ ms. The flow f_2 is thus accepted and the algorithm returns the new following matrix:

$$S^{\text{new}} = \begin{matrix} Nc & Nc & 0 & 0 & f_2 & 0 & 0 & 0 & 0 & 0 \\ Nc & Nc & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Nc & Nc & 0 & 0 & 0 & f_2 & 0 & 0 & f_1 & f_1 \\ Nc & Nc & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Nc & Nc & f_2 & 0 & 0 & 0 & f_1 & f_1 & 0 & 0 \\ Nc & Nc & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

This algorithm is a heuristic approach which aims at solving the problem formulated in Section 4.2. If the flow is not admitted, then the admission control stops; otherwise, the algorithm returns the new scheduling matrix S from which the gateway extracts the scheduling for every link of the path. The gateway creates a route response packet (RREP) to which it adds the scheduling for every link on the flow path.

4.3 Route selection

In this step, the gateway unicasts the RREP to the source. The RREP goes through the reverse path of the RREQ. Thus, every node along the path extracts and registers the reservation that the gateway has made for the links it belongs to as either a transmitter or a receiver. Once the RREP reaches the source node, the flow transmission can begin and every node along the path knows at which TUs it must forward the flow. Note that the selected TUs for a link are released if the starting node of the link does not receive any packet of the flow during a certain period p.

5 Simulation results

In this section, we conduct a simulation study using ns-2 [31] to evaluate and compare the performance of our proposed model, with the DCF MAC IEEE 802.11 standard. We evaluate several performance metrics like throughput, packet loss, and delay, under three different topologies: a chain topology, a grid topology, and a linear topology. Mesh nodes are distributed in a 1,000 m × 1,000 m coverage area. Every mesh node sends video flows to the gateway at a rate of 300 kbit/s and requires an end-to-end

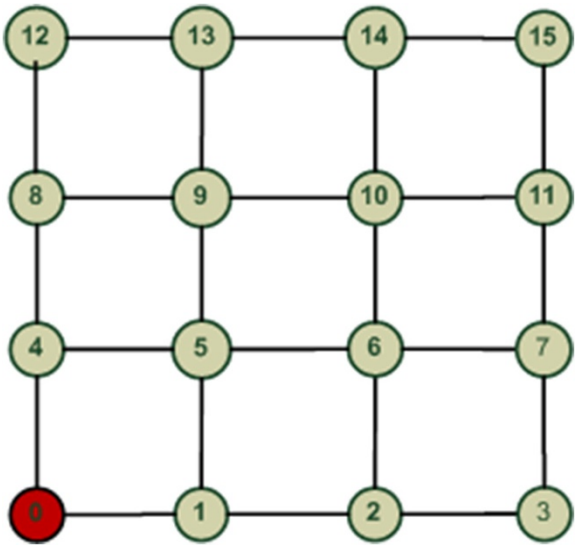

Figure 9 Grid topology.

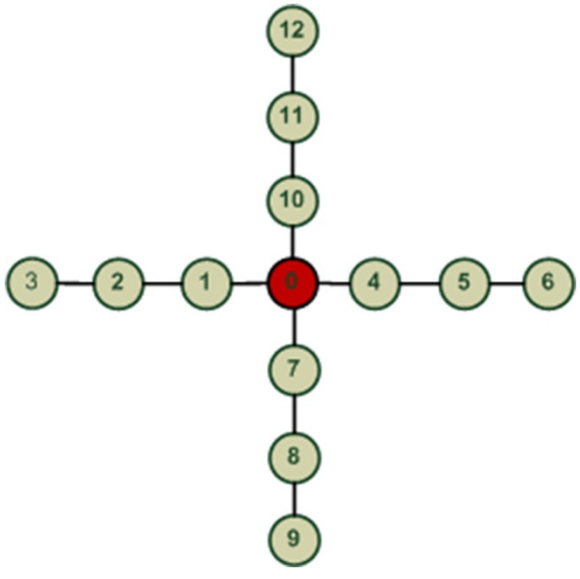

Figure 10 Cross topology.

Table 2 Simulation parameters

Layer	Parameters	Values
Signal propagation	Two-ray ground model	
Physical layer	PLCP preamble	20 µs
	Channel capacity	54 Mbit/s
MAC layer	TU interval	260 µs
	TS interval for the chain topology	30,160 µs
	TS interval for the grid topology	29,900 µs
	TS interval for the cross topology	30,160 µs
Transport layer	UDP size packet	1,000 bytes

delay of 150 ms. The flow identification represents the id of the source node and also its order of arrival in the network. Every node (except the gateway) in the network makes a request to admit a new flow in the network in the order of its source id; thus, node 0 is the first to make a request for a flow, then node 1, etc. Requests are sent periodically at 1 s of interval; node 0 first sends a request for flow 0, and a second after, node 1 makes a request for flow 1, etc. The chain topology is made up of 11 nodes, where the middle node (node 5) represents the gateway (see Figure 8). Thus, ten nodes in the linear topology make a request to admit a new flow. The grid topology is made up of 16 nodes (see Figure 9) and the cross topology is made up of 13 nodes (see Figure 10); node 0 represents the gateway for both topologies. Thus, 15 nodes in the grid network make a request to admit a new flow and 12 nodes in the cross topology.

Note that the length of TS interval is fixed in a way that it accepts many flows; thus, it varies according to the network topology. The different parameters used in simulations are presented in Table 2.

Figures 11, 12, 13, 14, 15 and 16 show the throughput of flows in the different topologies and models. With

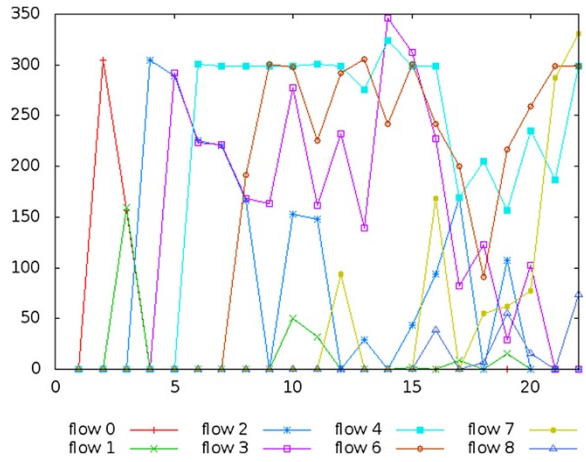

Figure 12 Flows' throughput in the IEEE model with a linear topology.

the linear topology, we can observe in our model (see Figure 11) that among the ten flows which make a request to be admitted in the network, nine flows are accepted and one is rejected. The flow which is rejected is the last to make its request; the network is then too loaded to accept it. The throughput of admitted flows stays quite stable over time, around 300 kbit/s; thus flows' requirements in terms of bandwidth are respected. In contrary, in the IEEE 802.11 model, the flows' throughput fluctuates and the 300 kbit/s required is not reached. In the cross topology, we can observe that our model (see Figure 13) accepts 10 flows among the 12 which make a request to be admitted. In the grid topology, we can observe that our model (see Figure 15) accepts 10 flows among the 15 which make a request to enter the network. The flows which are not accepted are sent by nodes situated far away from the gateway and which are the last to be sent in the network

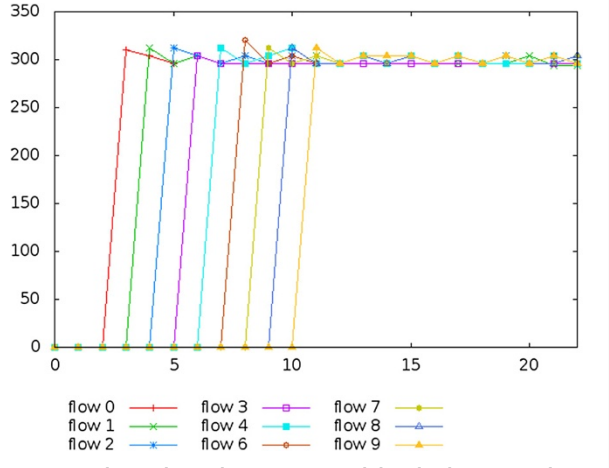

Figure 11 Flows' throughput in our model with a linear topology.

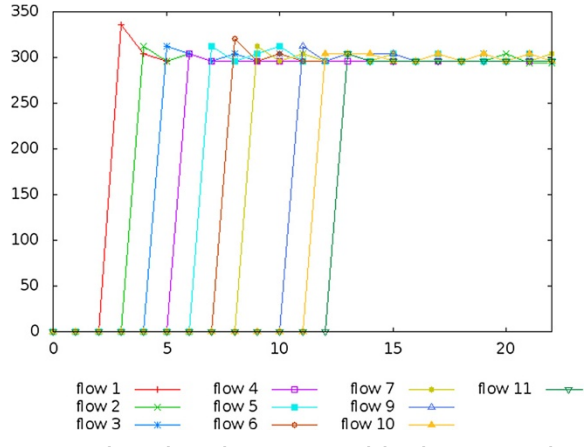

Figure 13 Flows' throughput in our model with a cross topology.

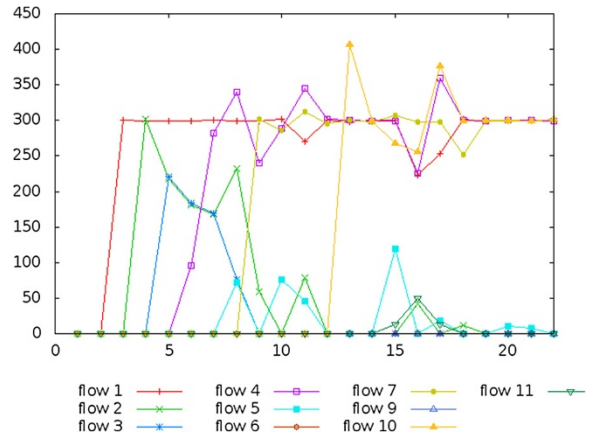

Figure 14 Flows' throughput in the IEEE model with a cross topology.

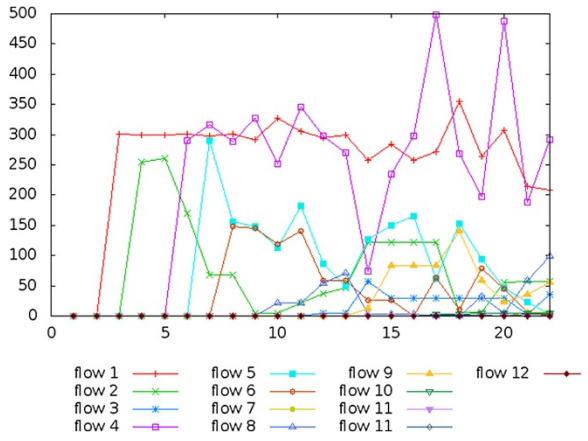

Figure 16 Flows' throughput in the IEEE model with a grid topology.

(i.e., flows 10, 12, 13, 14, and 15). As in the linear topology, our model in the cross and grid topologies meets the flows' requirements in terms of bandwidth and the flows' throughput remains quite stable over time, whereas the flows' throughput in the IEEE 802.11 fluctuates over time and a few of them get their required 300 kbit/s. Figures 17, 18, and 19 show the mean throughput for each flow in the different topologies. Some flows are not indicated on the figures as they are refused by our admission control model and none of their packets reach the gateway in the IEEE 802.11 model. In the chain topology (see Figure 17), all flows which are admitted in the network get the throughput they expected. We can observe that in the DCF IEEE 802.11 model, the more a flow is sent by a node situated far away from the gateway, the more its throughput is low; this phenomenon of starvation is well known and has already been analyzed in many papers (e.g., [32,33]). In the grid topology (see Figure 18), we can observe that in the IEEE 802.11 model, as in the previous topologies, the phenomena of starvation for flows are sent by nodes situated more than two hops away from the gateway whereas in our model, every admitted flow gets its required bandwidth. In the grid topology, as the gateway possesses only two neighboring nodes, only two nodes get almost the required bandwidth of 300 kbit/s (flows 1 and 4). In the cross topology (see Figure 19), as in the previous topologies, IEEE 802.11 suffers from starvation and our model reaches its goal to meet admitted flows' required bandwidth. Furthermore, as the gateway in the cross topology possesses only four neighboring nodes, four nodes get almost the required bandwidth of 300 kbit/s (flows 1, 4, 7, and 10). Our link scheduling scheme respects the throughput requirement for flows which are accepted in the network and shows an important gain compared to IEEE 802.11. This gain in throughput can be explained by many reasons such as:

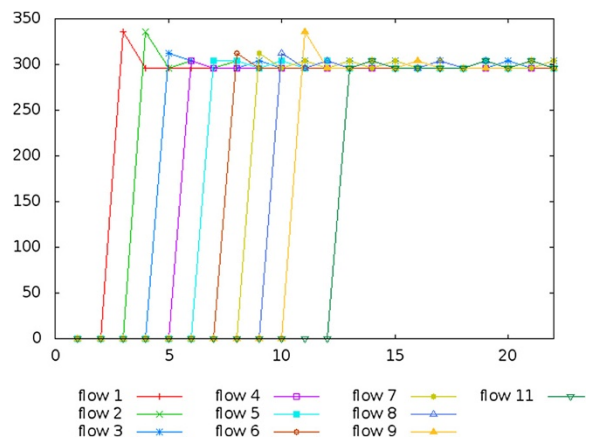

Figure 15 Flows' throughput in our model with a grid topology.

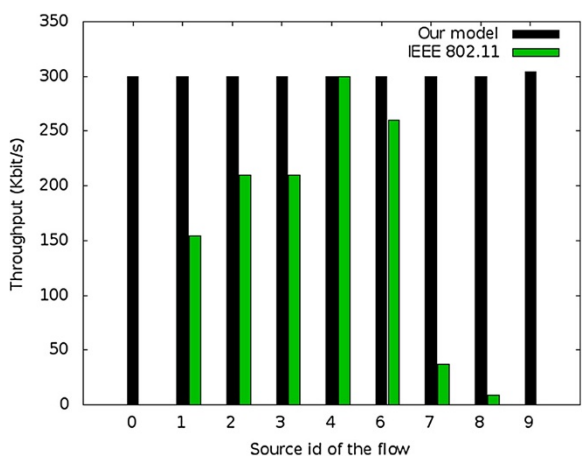

Figure 17 Throughput of the flows in the chain topology.

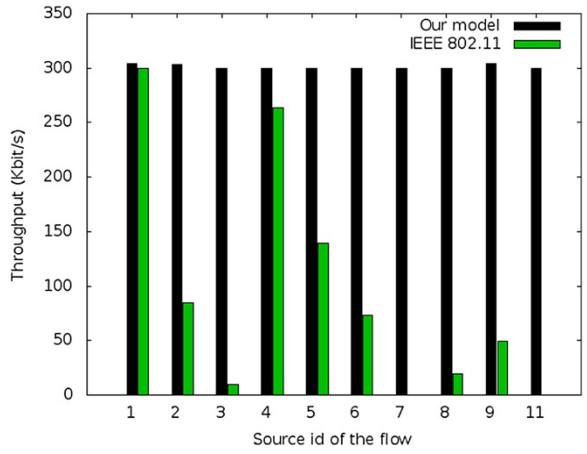

Figure 18 Throughput of the flows in the grid topology.

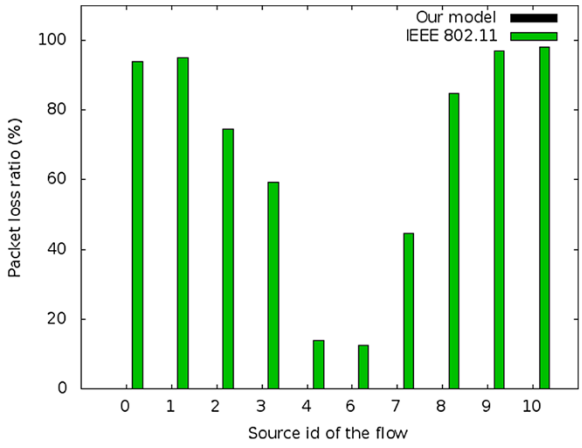

Figure 20 Packet loss ratio in a chain topology.

- The elimination of the RTS/CTS scheme. Indeed, as there is no risk of contention, the RTS/CTS scheme is not needed any more. Thus, the time used to send RTS or CTS packet in the IEEE 802.11 model can be exploited to send packets of data in our model.
- The elimination of the backoff procedure during the scheduling period. Indeed, because of the backoff algorithm in the IEEE 802.11 model, every node must wait a certain time, each time it wants to send a packet of data. The time that a node has to wait is chosen randomly in a contention window. The maximum contention window size is 1,023 slots [34] and a slot time lasts usually 20 μs; thus, a node can have to wait till 2,046 μs for sending a packet whereas sending a packet takes, in our example, 260 μs. Thus, eliminating the backoff procedure enables gaining a lot of time for sending packets of data.
- No risk of collision, thanks to the link scheduling. Because of collisions, in the IEEE 802.11 model, nodes need to resend packets; this leads to a waste of time which is avoided in our model.

Figures 20, 21, and 22 show the packet loss ratio of flows in the three different topologies. We can observe that our dynamic link scheduling reaches its goal of collision-free transmission for admitted flows. However, these results are obtained in an ideal state of simulation; indeed, packet loss may occur in a real environment. In IEEE 802.11, we can observe that the more the source of flow is situated far away from the gateway, the more the flow has a high packet loss ratio.

Figures 23, 24, and 25 show the flows' delay in the different topologies. The Y axis of these graphics possesses a logarithmic scale as the values of the flows' delay are very scattered. We observe in the grid topology (see Figure 24) that flows 10, 12, 13, 14, and 15 in our model possess no

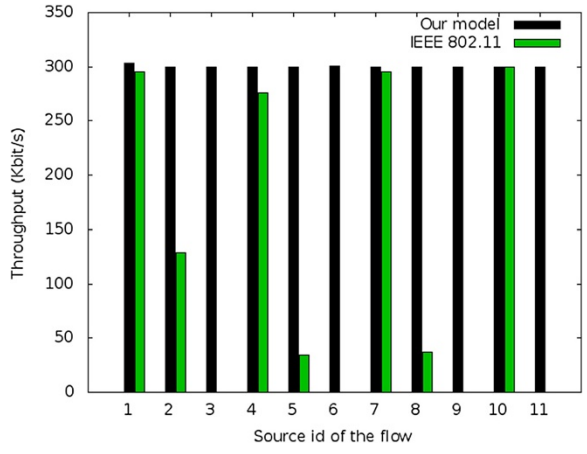

Figure 19 Throughput of the flows in the cross topology.

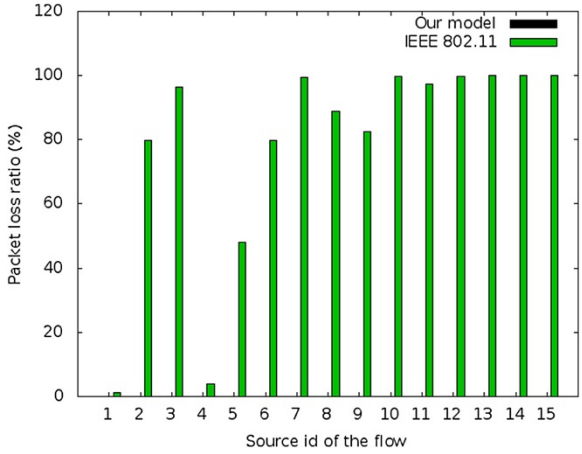

Figure 21 Packet loss ratio in a grid topology.

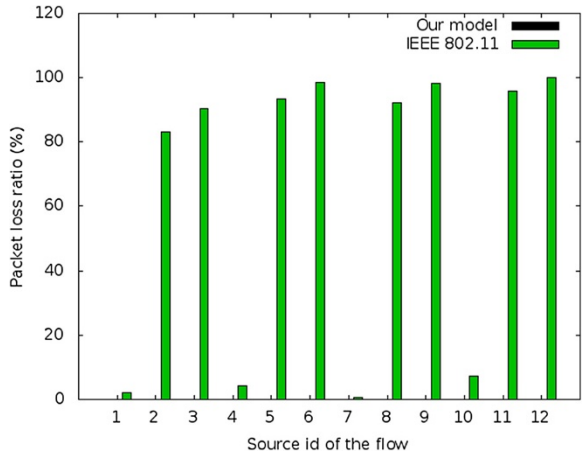

Figure 22 Packet loss ratio in a cross topology.

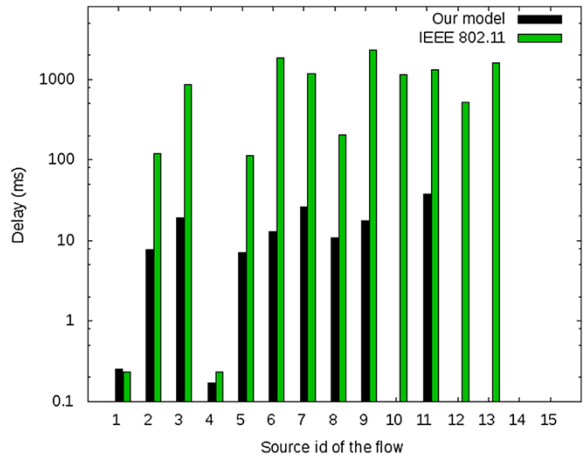

Figure 24 Flows' delay in the grid topology.

delay as these flows are not admitted in the network, and flows 14 and 15 in IEEE 802.11 possess no delay as none of their packets reach the gateway. For all these topologies, we can observe that our model satisfies the requirements of all flows in terms of delay as they get a delay inferior to 150 ms. For all the topologies, the IEEE 802.11 model expresses a delay inferior to 150 ms when their flow sources are situated at less than two hops away from the gateway; for the other flows, the obtained delay is superior to 150 ms.

Through these simulations, we can conclude that our dynamic link scheduling-based admission control model allows gaining in throughput, thanks to the dynamic time division multiple access (DTDMA) access method which enables getting rid of the backoff and RTS/CTS mechanisms. Furthermore, the delay and loss rate requirements are also satisfied compared to the IEEE 802.11 model. However, in our model, constraints are imposed, nodes must be synchronized, and gateways must know the power of the noise and the path loss of the signal at every moment.

6 Conclusion

In this paper, we have presented a new admission control scheme based on link scheduling to support real-time traffic in WMNs. We have considered both bandwidth and end-to-end delay as two major criteria in the design. The link scheduling is based on the SINR interference model in order to prevent any collision in flow packets. To enable a dynamic link scheduling, we have mixed two access methods: on one hand, CSMA/CA is used to send control packets (e.g., requests for flow admission) and on the other hand, DTDMA exploited for flow packet transmission, ensuring an important gain in throughput and a collision-free communication. Furthermore, we have introduced a method to compute the flow's delay and prove its efficiency. We have compared our model with the IEEE 802.11 model and shown under different topologies

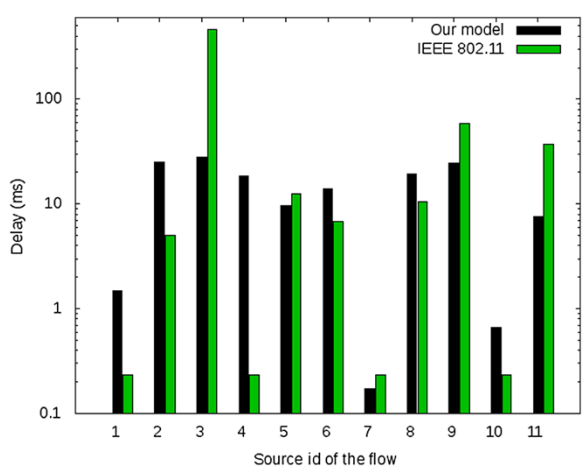

Figure 23 Flows' delay in the chain topology.

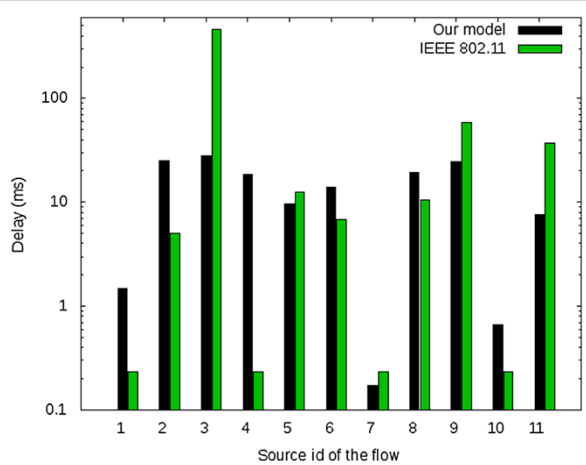

Figure 25 Flows' delay in the cross topology.

that our solution reaches its goal to respect admitted flow requirements in terms of delay and throughput. In a future work, we plan to compute the complexity of our model and compare it to other admission control schemes. Furthermore, we plan to secure the proposed model by integrating a robust trust mechanism in the backbone mesh. The study reported in this paper leaves several avenues open for further research on this intriguing problem. In particular, the problem of finding the length of optimal schedules without considering '*a priori* routing approach' remains open.

Competing interests
The authors declare that they have no competing interests.

Received: 28 February 2013 Accepted: 23 November 2013
Published: 20 December 2013

References

1. IF Akyildiz, X Wang, W Wang, Wireless mesh networks: a survey. Comput. Netw. **47**(4), 445–487 (2005)
2. J Jun, ML Sichitiu, The nominal capacity of wireless mesh networks. Wireless Commun. IEEE [see also IEEE Personal Communications]. **10**(5), 8–14 (2003)
3. Q Shen, X Fang, P Li, Y Fang, Admission control based on available bandwidth estimation for wireless mesh networks. Vehicular Technol., IEEE Trans. **58**, 2519–2528 (2009)
4. Y Yang, R Kravets, Contention-aware admission control for ad hoc networks. Mobile Comput., IEEE Trans. **4**, 363–377 (2005)
5. L Luo, M Gruteser, H Liu, D Raychaudhuri, K Huang, S Chen, A QoS routing and admission control scheme for 802.11 ad hoc networks, in *Proceedings of the 2006 Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks, DIWANS '06* (ACM New York, 2006), pp. 19–28
6. L Hanzo, R Tafazolli, Admission control schemes for 802.11-based multi-hop mobile ad hoc networks: a survey. IEEE Commun. Surv. & Tutorials. **11**, 78–108 (2009)
7. DM Shila, T Anjali, An interference-aware admission control design for wireless mesh networks. EURASIP J. Wireless Comm. Netw. **2010**, 7:1–7:13 (2010)
8. V Gambiroza, B Sadeghi, EW Knightly, End-to-end performance and fairness in multihop wireless backhaul networks, in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking, MobiCom '04* (ACM New York, 2004), pp. 287–301
9. K Papadaki, V Friderikos, Approximate dynamic programming for link scheduling in wireless mesh networks. Comput. & Oper. Res. **35**, 3848–3859 (2008)
10. G Brar, DM Blough, P Santi, Computationally efficient scheduling with the physical interference model for throughput improvement in wireless mesh networks, in *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking, MobiCom '06* (ACM New York, 2006), pp. 2–13
11. P Cappanera, L Lenzini, A Lori, G Stea, G Vaglini, Link scheduling with end-to-end delay constraints in wireless mesh networks, in *Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks & Workshops, 2009. WoWMoM 2009* (IEEE Piscataway, 2009), pp. 1–9
12. G Sharma, RR Mazumdar, NB Shroff, On the complexity of scheduling in wireless networks, in *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking, MobiCom '06* (ACM New York, 2006), pp. 227–238
13. K Sundaresan, HY Hsieh, R Sivakumar. Ad Hoc Netw. **2**, 109–132 (2004)
14. A Gore, A Karandikar, Link scheduling algorithms for wireless mesh networks. Commun. Surv. Tutorials, IEEE. **13**, 258–273 (2011)
15. J Dromard, L Khoukhi, R Khatoun, An admission control scheme based on links' activity scheduling for wireless mesh networks, in *Proceedings of the 11th International Conference on Ad-hoc, Mobile, and Wireless Networks, ADHOC-NOW'12* (Springer Berlin, 2012), pp. 399–412
16. J Rezgui, A Hafid, M Gendreau, *Distributed admission control in wireless mesh networks: models, algorithms, and evaluation*, vol. 59, (2010), pp. 1459–1473
17. DM Blough, G Resta, P Santi, Approximation algorithms for wireless link scheduling with SINR-based interference. IEEE/ACM Trans Netw. **18**, 1701–1712 (2010)
18. IEEE, *IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 10: Mesh Networking. IEEE Std 802.11s-2011* (IEEE, Piscataway, 2011), pp. 1–372
19. FR Vieira, JF de Rezende, VC Barbosa, S Fdida, Scheduling links for heavy traffic on interfering routes in wireless mesh networks. Comput. Netw. **abs/1106.1590**, 1584–1598 (2012)
20. K Xu, M Gerla, S Bae, How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks, in *Proceedings of the Global Telecommunications Conference, 2002. GLOBECOM '02, vol. 1* (IEEE Piscataway, 2002), pp. 72–76
21. K Jain, J Padhye, VN Padmanabhan, L Qiu, Impact of interference on multi-hop wireless network performance, in *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking, MobiCom '03* (ACM New York, 2003), pp. 66–80
22. A Gore, A Karandikar, S Jagabathula, On high spatial reuse link scheduling in STDMA wireless ad hoc networks, in *Proceedings of the IEEE Global Telecommunications Conference, 2007. GLOBECOM '07* (IEEE Piscataway, 2007), pp. 736 –741
23. DM Blough, G Resta, P Santi, Approximation algorithms for wireless link scheduling with SINR-based interference. IEEE/ACM Trans. Netw. **18**, 1701–1712 (2010)
24. P Cappanera, L Lenzini, A Lori, G Stea, G Vaglini, Optimal joint routing and link scheduling for real-time traffic in TDMA wireless mesh networks. Comput. Netw. **57**, 2301–2312 (2013)
25. I Rhee, A Warrier, J Min, DRAND: distributed randomized TDMA scheduling for wireless ad hoc networks, in *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc '06* (ACM New York, 2006), pp. 190–201
26. A Iyer, C Rosenberg, A Karnik, What is the right model for wireless channel interference? Wireless Commun. IEEE Trans. **8**, 2662–2671 (2009)
27. YB Bai, X Zhu, X Shao, WT Yang, FAST: fuzzy decision-based resource admission control mechanism for MANETs. Mobile Netw. Appl. **17**, 758–770 (2012)
28. A Krasilov, A Lyakhov, A Safonov, Interference, even with MCCA channel access method in IEEE 802.11s mesh networks, in *Proceedings of the 2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems, MASS '11* (IEEE Washington, DC, 2011), pp. 752–757
29. L Lenzini, E Mingozzi, C Vallati, A distributed delay-balancing slot allocation algorithm for 802.11s mesh coordinated channel access under dynamic traffic conditions, in *Proceedings of the 2010 IEEE 7th International Conference on Mobile Adhoc and Sensor Systems (MASS)* (IEEE Piscataway, 2010), pp. 432–441
30. P Gupta, PR Kumar, The capacity of wireless networks. IEEE Trans. Inf. Theory. **46**, 388–404 (2000)
31. The Network Simulator - ns-2. http://www.isi.edu/nsnam/ns/. Accessed 11 Dec 2013
32. A Lyakhov, I Pustogarov, A Safonov, M Yakimov, Starvation effect study in IEEE 802.11 mesh networks, in *Proceedings of the IEEE 6th International Conference on Mobile Adhoc and Sensor Systems, 2009. MASS '09* (IEEE Piscataway, 2009), pp. 651 –656
33. O Gurewitz, V Mancuso, J Shi, EW Knightly, Measurement and modeling of the origins of starvation of congestion-controlled flows in wireless mesh networks. IEEE/ACM Trans. Netw. **17**, 1832–1845 (2009)
34. IEEE, *IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)* (IEEE, Piscataway, 2007), pp. 1–1076

doi:10.1186/1687-1499-2013-288
Cite this article as: Dromard *et al.*: An efficient admission control model based on dynamic link scheduling in wireless mesh networks. *EURASIP Journal on Wireless Communications and Networking* 2013 **2013**:288.