Telecommun Syst (2017) 66:39–54
DOI 10.1007/s11235-016-0273-0

Towards combining admission control and link scheduling in wireless mesh networks

J. Dromard[1] · **L. Khoukhi**[1] · **R. Khatoun**[2] · **Y. Begriche**[2]

Published online: 9 January 2017

Abstract Wireless mesh networks (WMNs) have emerged recently as a key solution for next-generation wireless networks; they are low cost and easily deployed technology. However, WMNs have to deal with a low bandwidth which prevents them from guaranteeing the requirements of applications with strict constraints. To overcome this limitation, we propose in this paper a new admission control model which integrates a dynamic link scheduling scheme, named ACLS, in order to optimize the network bandwidth use. We formulate the admission control problem as a binary linear programming problem (BL2P). The proposed admission control integrates an algorithm, based on the Dakin's branch and bound (B&B) method, which respects the bandwidth and delay required by the flows. The proposed ACLS solution has been validated on ns2, and the simulation results showed that ACLS model has better performance than the reference solution BRAWN; it accepts more flows while guaranteeing their delay and bandwidth.

Keywords Admission control · Quality of service · Wireless mesh networks · Link scheduling

✉ R. Khatoun
rida.khatoun@telecom-paristech.fr

J. Dromard
dromardj@utt.fr

L. Khoukhi
lyes.khoukhi@utt.fr

Y. Begriche
youcef.begriche@ieee.org

[1] University of Technology of Troyes, 12 Rue Marie Curie, BP 2060, 10010 Troyes, France

[2] Telecom Paristech, 46 Rue Barrault, 75013 Paris, France

1 Introduction

Wireless mesh networks (WMNs) introduce new opportunities for wireless networks especially in smart cities [1] where infrastructures utilizing thousands of sensors. Indeed, they can, for a low cost, rapidly extend Internet access in areas where cables' installation is impossible or economically not sustainable such as hostile areas, battlefields, old buildings, rural areas, etc. [2]. A WMN is made up of fixed mesh routers (MRs) which relay the information of mesh clients (MCs) from MR till reaching a gateway. Most WMNs' studies assume that every node has a single antenna 802.11-based [3–7]. Furthermore, IEEE has recently developed a set of standards for WMNs (e.g., the IEEE 802.11s) [8], where every node possesses a WiFi antenna [9]. In our study and hereinafter, we only consider single antenna 802.11-based WMNS, as it seems to be the most widely spread technology. However, the lack of resources in these networks in terms of bandwidth, limits their deployment. Indeed, the lack of bandwidth, due mainly to the contention access to the channel, may lead to unfairness between nodes and congestion [7,10,11]. For example, in [12], the authors show that, when the number n of nodes increases, the nodes' throughput of WMN decreases in $O(1/n)$. Thus, WMNs may often not respect the requirements of flows which have strict constraints. To solve this issue, many admission control (AC) schemes have been proposed for WMNs [13–17].

An AC scheme aims at accepting a new flow in the network only if its requirements and the requirements of still-active previously admitted flows can be met [18–20]. However, these solutions are limited in the number of flows they can accept due to the WMNs' lack of bandwidth. In order to increase WMN's capacity, multiple-antenna systems have been proposed [21–24]. However, there are several reasons for not using multi-antenna nodes; like the cost and size of

nodes which must be often kept small and the minimum required distance between the antennas of a multi-antenna system [25]. Indeed, to have effective antennas on the same node, they must be separated by a distance of more than half a wavelength [25,26].

Thus, to improve WMNs throughput, we focus on link scheduling (LS) schemes rather than multi-antenna systems. In [27], the authors show, via simulations, that using an LS scheme rather than IEEE 802.11 scheme can increase up to three times a WMN's throughput. LS scheme aims at assigning, to each link, the slots during which a node can send data without any risk of collision while maximizing the network throughput. However, the scheduling often does not evolve in time and with the network load. Indeed, a fixed link scheduling may lead to a loss in the WMN throughput use. Slots, which are not used by the links to which they are assigned, are lost while they could be used by some other links.

In order to take advantage of both AC and LS schemes, we propose in this paper a new AC which integrates LS scheme. Our idea is to make the network's LS evolves each time a new flow is admitted or leaves the network. The aim of our solution is to respect the delay and bandwidth of every admitted flow while increasing the number of accepted flows compared to existing admission controls. We note that the MAC protocol is changed to a synchronous time-based link scheduling mechanism; only the physical layer of IEEE 802.11 standard can be used. The contributions of this paper can be summarized as follows:

- A modeling of the AC with LS problem as a binary linear programming problem (BL2P).
- A new algorithm which solves the admission control with LS problem and which is based on the Dakin's branch and bound (B&B) method [28]. This algorithm is, in the following, denoted ACL2S for Admission Control with Link Scheduling Solver.
- A complete solution of admission control with link scheduling in a WMN, named ACLS.

To the best of our knowledge, our proposed ACLS is one of the first works integrating two separate concepts: admission control and link scheduling. The remainder of the paper is organized as follows. In the following section, we present a brief state of the art in the admission control and link scheduling fields. In Sect. 3, we propose a modeling of the WMN. Then, the admission control problem with link scheduling is modeled as a binary linear programming problem. In Sect. 5, we introduce a new method to compute flows' delay, knowing their scheduling. Then, we present ACL2S, an iterative algorithm based on Dakin's B&B method which solves our admission control problem. In Sect. 7, we describe the implementation of our admission control with link scheduling. Finally, we display the results obtained via simulations on ns2 and then we conclude the paper.

2 Related work

In this section, we present related work in both link scheduling and admission control fields. We also introduce some existing works joining both admission control and link scheduling concepts. Link scheduling schemes allows to guarantee collision-free transmissions and increase the bandwidth use [27]. They select, for each link in the network, the slots in a frame during which the link is periodically activated. The selection process aims at ensuring interference-free transmission and at maximizing the network throughput [29]. To avoid collisions, LS schemes employ an interference model in order to identify the links that could be activated simultaneously without causing any interference issue. The problem of maximizing the throughput in WMNs using the link scheduling is known as a NP-hard, even with a simple interference model [30]. LS schemes can be classified, according to the interference model they are based on. It is usually either (1) a hop-based (e.g., [30,31]), or (2) a distance-based (e.g., [32,33]), or (3) a SINR-based interference model (e.g., [27,34–36]).

With 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. The authors in [27,31,37] propose LS scheme based on the hop interference model. In [27], the authors formulate the k-hop interference model as a k-valid matching problem using graph theory. The authors develop LS scheme based on a greedy algorithm which computes sets of independents maximum k-valid matching in the network graph. A maximum k-valid matching is the maximum set of edges which are at least k-hops from each other. According to the k-hop interference model, the nodes of such a set can be activated simultaneously without any risk of interference. Nonetheless, this model can only be deployed to a limited number of topology.

In a distance-based model, a transmission succeeds if the distance between the transmitter and the receiver is smaller than or equal to the communication range R_c and if no other node is transmitting in the interference range R_i of the receiver [32,35] (see Fig. 1).

In [33], the authors develop new models for calculating upper and lower bounds on the optimal throughput for any given network and propose an algorithm to schedule links. Nevertheless, they assume that packet transmissions at the individual nodes can be controlled and scheduled by an omniscient central entity, which is hard to achieve.

With the SINR-based interference model, a node can successfully receive data if its SINR is greater than or equal to

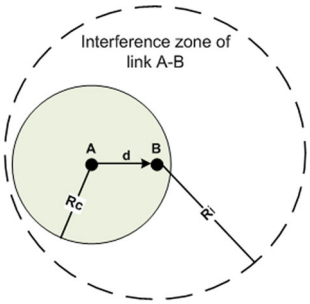


Fig. 1 It shows the communication range R_c and the interference range R_i of a link A–B using the distance based interference model

a threshold which value depends principally on the physical layer properties of the network card [29].

In [27], the authors propose a link scheduling model based on a centralized polynomial time algorithm using the SINR-based model interference. This algorithm schedules link by link; each link is scheduled at slots such that the resulting sets of scheduled transmission is feasible. To maximize the network throughput, this algorithm aims at minimizing the *schedule length* (i.e., at finding the shortest frame which allows to schedule every link). However, the authors assume that flows' requirements are known a priori by the scheduling module, which is an unrealistic assumption.

Some studies show that hop-based and distance-based interference models allow low computation; nevertheless, they can both accept flows that lead to interference and may reject other flows that are interference-free [35,38]. Many studies show that the SINR-based interference model is the most accurate interference model; however, it is also the more complex one [35,38]. In the works presented above, as in most of existing works on link scheduling, the scheduling is done in a static way. This means that the number of slots, dedicated to each link, does not evolve in time and with the network load. Thus, whether the network is loaded or not, a link has always the same number of reserved slots which can lead to congestion issues and a non-optimal bandwidth use.

An admission control scheme aims at accepting a new flow in the network only when it can guarantee its requirements (often delay and bandwidth) and the requirements of previously admitted flows. To decide whether a flow can be accepted, an AC scheme evaluates whether each node, along the path, can meet the new flows' requirements. If it is the case, it accepts the new flow; otherwise, it rejects it. One of the distinctive features of an AC scheme is its method to compute the available bandwidth of nodes [39]. In the AC models developed in [6,39–41], the authors reported that a node's available bandwidth depends mainly on its channel idle time ratio (CITR). A node's CITR is equal to the fraction of idle time of its carrier sensing range multiplied by the capacity of its channel. However, considering only the CITR may lead to an inaccurate estimation of the available bandwidth. Indeed, sending a packet when a node's channel is idle does not guarantee an interference free transmission due mainly to hidden nodes.

To overcome this issue, the authors in [41] propose a probabilistic approach to estimate the available bandwidth of a node which does not trigger any overhead and which considers the node's CITR and hidden terminals. Upon this available bandwidth estimation, an Admission Control Algorithm (ACA) is developed to differentiate QoS levels for various traffic types.

In [42], the authors design a fuzzy decision-based multicast routing resource admission control mechanism (FAST). In FAST, once every node on a flow path has accepted the flow in terms of bandwidth, the source node takes the final decision to accept or not the flow according to the result of a fuzzy-decision scheme. The fuzzy-decision scheme considers delay, jitter, packet loss, and bandwidth. The validation of this solution shows good performance; nevertheless, the authors do not explain how they obtain the values of the QoS parameters used in their fuzzy decision scheme.

In [43], the authors propose IAC, an Interference-aware Admission Control for WMNs. The originality of IAC lies in a dual threshold based method to share the bandwidth between neighboring nodes. However, this method cannot deal with multi-rate nodes and does not consider the option of parallel transmissions which may lead to an under-estimation of the nodes available bandwidth.

It is worth noting that most of the proposed AC schemes are based on CSMA/CA standard. The latter is well known to lead to poor throughput [27]. Indeed, CSMA/CA triggers interference and dedicates a huge amount of time to avoid collision (via the backoff algorithm and the RTS/CTS mechanism). To overcome these issues, few solutions integrate both AC and LS schemes [8,18,44,45].

For example, the IEEE 802.11s standard [8] proposes the protocol mesh coordinated channel access (MCCA). 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. However, MCCA may suffer from collisions due to the hidden nodes problem [4] and does not specify any link scheduling algorithm [3].

In a previous work [44], we proposed to integrate link scheduling in an admission control scheme. The link scheduling scheme we proposed, is totally distributed among the WMN's nodes and is based on the distance-based interference model. A flow is admitted, if there exists a path ensuring that every node is able to compute a link scheduling for this flow while guaranteeing its demands in terms of bandwidth. However, this solution generates an important overhead due to the broadcast of advertisement packets. In [18], we proposed a solution which integrates both an AC and LS scheme based on SINR-based interference model. In this solution, the

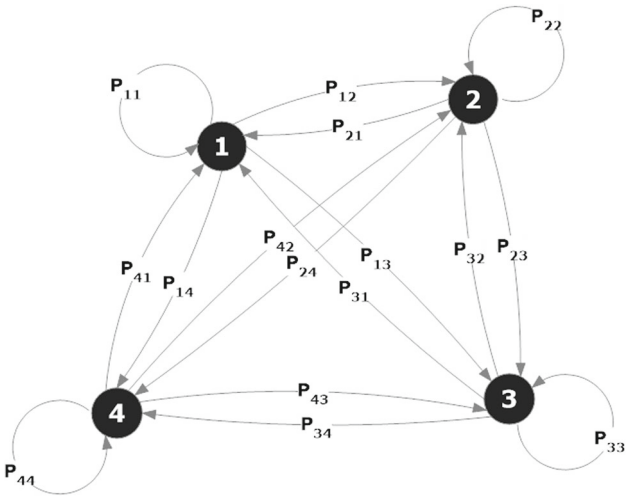


Fig. 2 Power graph

AC is performed at the gateway. When a flow is accepted, the gateway informs nodes on the flow path of their new scheduling. However, this solution lacks of a theoretical background; furthermore, it has not been evaluated and compared to existing AC solutions.

3 Network model

We represent the WMN by a power graph (see Fig. 2). The power graph is denoted $G'(V, E, f)$, where V represents the set of nodes in the WMN, E is the set of directed links in the network and f is a function. Each node u has a directed link to every node $v \in V - \{u\}$ in the network denoted (u, v), where u is said to be the sender and node v the receiver. Every node $u \in V$ has also a loop denoted (u, u). The function $f : E \rightarrow \mathbb{R}$ associates to each link $e = (u, v)$, a value which represents, when $u \neq v$, the power P_{uv} at which v receives a signal from u, and when $u = v$, the noise power N_u at the node u. This graph can be represented by a matrix P of size $|V| * |V|$ where each element:

$$p_{ij} = \begin{cases} P_{ij} & \text{if } i \neq j \\ N_i & \text{if } i = j \end{cases}$$

We introduce also two other matrices, which are both of size $|E| \cdot |V|$; the senders' matrix L^s and the receivers' matrix L^r. Each element l^s_{ij} of the senders' matrix is equal to:

$$l^s_{ij} = \begin{cases} 1 \text{ if node } j \text{ is the sender of the link } e_i \\ 0 \text{ if node } j \text{ is not the sender of the link } e_i \end{cases}$$

Each element l^r_{ij} of the receivers' matrix is equal to:

$$l^r_{ij} = \begin{cases} 1 \text{ if node } j \text{ is the receiver of the link } e_i \\ 0 \text{ if node } j \text{ is not the receiver of the link } e_i \end{cases}$$

With the previous introduced matrices, we can compute the power P_{vu} at which a node u receives a signal from a node v, knowing that v sends its data over the link $e_i = (v, z)$ and node u is link e_j's receiver:

$$P_{uv} = l^s_{i.} \cdot P \cdot t(l^r_{j.}) \tag{1}$$

$l^r_{j.}$ represents the jth row of the matrix L^r, and $l^s_{i.}$, the ith row of the matrix L^s. Furthermore, $t(l^r_{j.})$ is the transpose of the vector $l^r_{j.}$.Thereafter, we denote t(A) the transpose of a matrix or a vector A. We assume that time is divided in frames made up of N slots. There are N_c slots of competition and N_s slots of scheduling with $N = N_c + N_s$. The first N_c slots of a frame are used to send control packets. The access mode used for packets of control is based on competition. The N_s following slots are used to send data packets, and the access mode is based on scheduling. We assume that all data packets have the same length and are sent at the same frequency. Thus, every packet of data and their acknowledgment (ACK) have the same transmission time. A length of a slot is equal to the time of transmission of a packet of data plus its acknowledgment (ACK).

Each flow f requires a certain delay and bandwidth. A flow is admitted along a path p_f if a scheduling, respecting the flow's delay and bandwidth, is found. Thus, a flow's requirements is associated to a triplet $a_f = (N_f, d^f, p_f)$, where N_f represents the number of slots per frame which are required by the flow to respect its bandwidth, and d^f is the delay the flow requires. The path p_f of a flow f is an ordered list of links, where the ith element of the list is the ith link on the flow path. When a flow is admitted along a path p_f, it is then associated to a scheduling matrix, denoted S^f of size $|E| \cdot |N_s|$. This matrix indicates for every link on the flow f path the slot(s) during which it can send the flow. An element s^f_{ij} of a matrix S^f is equal to:

$$s^f_{ij} = \begin{cases} 1 & \text{if the } (N_c + j)\text{th slot is reserved by} \\ & \quad \text{link } e_i \text{ to send a packet of flow } f \\ 0 & \text{if the } (N_c + j)\text{th slot is not reserved by} \\ & \quad \text{link } e_i \text{ to send a packet of flow } f \end{cases}$$

The network scheduling is represented by the matrix S of size $|E| \cdot |N_s|$. This matrix is the sum of the scheduling matrix of every flow f admitted and still active in the WMN, thus:

$$S = \sum_{\forall f \in F} S^f \tag{2}$$

with F the set of admitted and still active flows in the network. In the following, each element of matrix S is denoted s_{ij}.

4 Modeling of the admission control with link scheduling problem as a BL2P

A flow f is admitted along a path p_f if a matrix S^f can be found which respects a set of constraints among which the flow requirements described by the triplet $a_f = (N_f, d^f, p_f)$. In the following, we present the six different constraints the matrix S^f must respect. First, the matrix S^f must respect the flow constraint in terms of number of slots per frame required by the flow, and therefore the following constraint:

$$\forall i, e_i \in p_f, N_f = \sum_{j=1}^{j=N_s} s_{ij}^f \tag{3}$$

Furthermore, a slot can only be scheduled once by a link. Therefore a link e_i can be, for a slot j, either scheduled (then $s_{ij} = 1$) or not (then $s_{ij} = 0$). The second constraint can be expressed by the following formula:

$$\forall i, e_i \in E \; and \; \forall j \in [1, N_s], s_{ij} + s_{ij}^f \leq 1 \tag{4}$$

Furthermore, every link, not belonging to the path of the flow, does not need to reserve any slot for the flow; thus, the matrix S^f must also respect the third following constraint:

$$\forall i, e_i \notin p_f \; and \; \forall j \in [1, N_s], s_{ij}^f = 0 \tag{5}$$

During each slot reserved by a link (u, v), a packet of data is sent from u to v and an acknowledgment is then sent from v to u. As many research works (e.g., [38,46]) have shown that the SINR model is the most realistic interference model, we have decided to apply this model in our work. A node u can transmit with success, according to the SINR model, a packet to a node v, while there is a set Γ of simultaneously activated links in the network, if:

$$\frac{P_{uv}}{N_v + \sum_{\forall (w,z) \in \Gamma - \{(u,v)\}} P_{wv}} \geq \beta \tag{6}$$

β represents the SINR threshold, its value depends mainly on the physical layer technique and the decoding strategy used [47]. The links must be scheduled so that no link interferes with another according to the SINR model. To reach this goal, at every slot j, the power at which the receptor of a link e_i, scheduled at slot j receives the signal from the link e_i's sender, divided by the sum of the noise and the interference at the receptor, must be superior to the SINR threshold. The power, at which the receptor u of a link e_i receives the signal from its sender v, is equal to p_{uv} (the element of index uv of the power matrix P). According to Eq. 1, this power equals to $l_{i.}^s \cdot P \cdot t(l_{i.}^r)$. Furthermore, the noise at the node u equals to p_{uu}, and therefore to the element of index uu of the power matrix P. The noise can consequently be obtained by the following formula: $l_{i.}^r \cdot P \cdot t(l_{i.}^r)$. Furthermore, a link sends data at a slot j if it is scheduled at that slot, i.e., if $s_{ij} + s_{ij}^f = 1$. The interference at link e_i's receptor equals to the sum of the powers at which its receptor receives the signal from all the senders of the links which have reserved the same slot j, i.e., the set of links $e_z \in E$ such as $s_{zj} + s_{zj}^f = 1$ minus the link e_i. The interference at the receptor of link e_i can be thus expressed as follows: $\sum_{\forall z, e_z \in E - \{e_i\}} (s_{zj} + s_{zj}^f) \cdot l_{z.}^s \cdot P \cdot l_{i.}^r$. Thus, the matrix S^f must respect a fourth constraint expressed by Eq. 7 so that every packet of data is received successfully:

$$\forall i, e_i \in E \; and \; j \in [1, N_s] \quad \frac{(s_{ij} + s_{ij}^f) \cdot l_{i.}^s \cdot P \cdot t(l_{i.}^r) + \Lambda \cdot (1 - (s_{ij} + s_{ij}^f))}{l_{i.}^r \cdot P \cdot t(l_{i.}^r) + \sum_{\forall z, e_z \in E - \{e_i\}} (s_{zj} + s_{zj}^f) \cdot l_{z.}^s \cdot P \cdot l_{i.}^r} \geq \beta \tag{7}$$

Λ is a large positive integer which magnitude will be discussed later in this paper. It is introduced so that, when a slot j is not reserved by a link e_i ,the formula remains true. Furthermore, the receptor of every link e_i must also send an acknowledgement to the sender after having successfully received a packet of data. To insure that acknowledgement arrives successfully at the sender of link e_i, the matrix S^f must also respect a fifth constraint expressed by the following inequality 8.

$$\forall i, e_i \in E \; and \; j \in [1, N_s] \quad \frac{(s_{ij} + s_{ij}^f) \cdot l_{i.}^r \cdot P \cdot t(l_{i.}^s) + \Lambda \cdot (1 - (s_{ij} + s_{ij}^f))}{l_{i.}^s \cdot P \cdot t(l_{i.}^s) + \sum_{\forall z, e_z \in E - \{e_i\}} (s_{zj} + s_{zj}^f) \cdot l_{z.}^r \cdot P \cdot l_{i.}^s} \geq \beta \tag{8}$$

In Inequalities 7 and 8, β is the SINR threshold of both nodes i and j. We assume here, for simplicity, that their SINR threshold is equivalent. However, our solution stays true even with nodes with different SINR threshold. Furthermore, as explained previously, Λ is introduced so that, when a slot j is not reserved by a link e_i the formulas stay true. Λ is a large positive number which magnitude is estimated as follows. It must be larger than the product of β and the sum of the noise and the interference noise at the link e_i's receiver for the Inequality 8 and the link ei's receptor for the Inequality 7. The magnitude of Λ depends mostly on the value of the nodes' physical layer. If we assume that the power at which each node sends a packet is about 100 mW, that the SINR is about 10 dbm and that there is, at most, 100 nodes active simultaneously and if we don't consider the attenuation due to the distance and the thermal noise, which is often small compared to the nodes' transmit power, Λ can be set to $10^5 (100 * 100 * 10)$. However, to insure that both equations

stay always true, we advise a margin of magnitude 10 and so a Λ of at least 10^6.

A node $v \in V$ cannot be the sender and the receiver of a link. Furthermore, a node $v \in V$ can be the sender or the receiver of many links; however, these links cannot be scheduled simultaneously, i.e., at the same slot j. For reminder, a link e_i is scheduled at slot j if $s_{ij} + s_{ij}^f = 1$ otherwise $s_{ij} + s_{ij}^f = 0$ and $v$is the receiver or the sender of a link e_i if $l_{iv}^s + l_{iv}^r = 1$. As every node v cannot be simultaneously the receiver and the sender of a link and as no link scheduled at the same slot j cannot possess a node in common, the matrix S^f must respect a sixth constraint expressed by the following formula:

$$\forall v \in V \text{ and } \forall j \in [1, N_s], \sum_{i=1}^{i=|E|} (s_{ij}+s_{ij}^f)\cdot(l_{iv}^s+l_{iv}^r) \leq 1 \quad (9)$$

Finally, finding a scheduling for a new flow f which respects its required bandwidth along a path p_f, when the delay constraint is not taken in account, can be formulated as a BL2P which objective is to find the elements of the matrix of scheduling S^f which minimizes the number of reserved slots. The BL2P can be expressed as follows:

$$Minimize \sum_{i=1}^{i=|E|} \sum_{j=1}^{j=N_f} s_{ij}^f \text{ with } s_{ij}^f \text{ a binary variable} \quad (10)$$

while respecting the constraints represented by Eqs. 3, 4, 5, 7, 8 and 9. Equation 10 is the objective function of the BL2P, and the value obtained with this function, for a given matrix S^f, is the objective value.

5 Delay of a flow

This section is divided in four parts. In the first one, we propose an algorithm to compute the queue length of each intermediate node on the flow path at the beginning of every frame. In the second part, we introduce a second algorithm to compute the queue length of every intermediate node at any slot of any frame $m > 1$. In a third part, we propose a method to compute the delay of a packet at a node. Finally, in a fourth part, we introduce a method to compute a flow's delay.

The path $p_f = (l_1 \ldots, l_n)$ of a flow f is made up of an ordered set of links which belong to E. They are ordered such that every packet is first sent over l_1 then over l_2, etc. We denote u_i, link l_i's transmitter. We assume that each intermediate node on the flow path has a FIFO queue for the flow. Each link $l_i \in p_f$ is associated to a N_f-uplet denoted $\Psi^{f,i}$, where each element $\Psi_j^{f,i}$ ($j \in [1, N_f]$) represents the slot's number of the jth reserved slot of node u_i; these slots are listed in an increasing order. For example, in Fig. 3, node u_i has reserved the slots number 1, 5 and 7 to send packets of flow f; thus, $\Psi^{f,i} = (1, 5, 7)$, $\Psi_1^{f,i} = 1$, $\Psi_2^{f,i} = 5$, $\Psi_3^{f,i} = 7$ and $N_f = 3$.

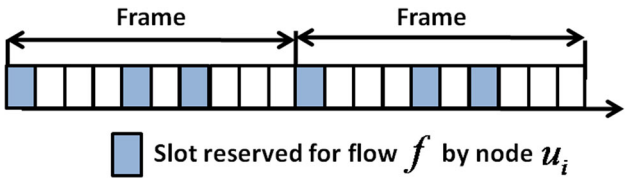


Fig. 3 Node u_i has reserved the slots number 1, 5 and 7 for the flow f

As we want to maximize the flows delay, we assume that the rate is maximum. The rate is maximum when the source node sends continuously a packet each time it can, i.e., at each slot it is authorized to send a packet. Thus, to compute the maximum flows delay, we make the two following assumptions:

- Every node receives a packet at each slot reserved by its previous node on the flow path.
- Every node, at the beginning of the first frame, does not possess any pending packets of flow f.

Furthermore, we introduce the following two definitions:

- A packet's delay is the sum of its delay at each node on the flow's path.
- The delay of a flow is the largest delay reached by one of its packets.

5.1 Node's queue length at the beginning of a frame

In the following, we denote by $q_{m,z}^{f,i}$, the number of packets of flow f that node u_i has in its queue at the beginning of the slot z (or zth slot) of the mth frame. For example, we note $q_{2,1}^{f,i}$ the queue's length of an intermediate node u_i at the beginning of the first slot of the second frame.

Each node on the path of a flow f, except the destination, possesses N_f slots scheduled for f. Thus, every intermediate node can forward all the N_f packets it receives from a neighbor during a frame before the end of the next one. It can be concluded that no node possesses more than N_f pending packets of f in its queue, and that every node can forward a packet during either the current or the next frame. For example, in Fig. 4, we can notice that node u_i receives two packets during the first frame. At the end of the first slot of the second frame, it has forwarded both.

If node u_i's queue length is equal to $q_{2,1}^{f,i}$ at the beginning of the second frame, it implies that among the N_f packets that u_i receives during the first frame, it forwards $N_f - q_{2,1}^{f,i}$

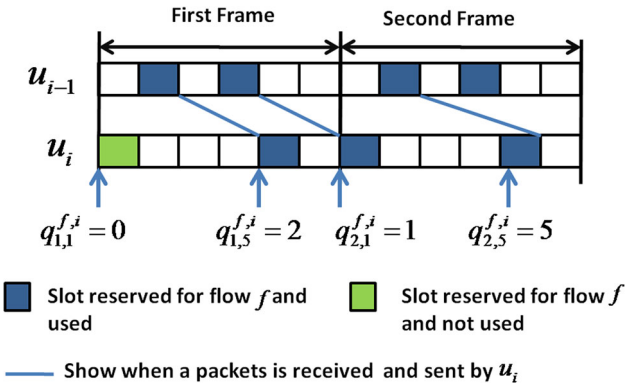


Fig. 4 Node u_i and u_{i-1} reserved slots for flow f and $N_f = 2$

of them. It also implies that the $N_f - q_{2,1}^{f,i}$ last reserved slots of u_i are each situated after a different reserved slot of u_{i-1}. That is why the $N_f - q_{2,1}^{f,i}$ first packets received by u_i during the first frame can be all forwarded before the end of the current frame. During the second frame, every intermediate node u_i on the flow path receives N_f new packets and sends, during its first $q_{2,1}^{f,i}$ reserved slots, the $q_{2,1}^{f,i}$ pending packets it had at the beginning of the frame. It has then still $N_f - q_{2,1}^{f,i}$ last reserved slots available in the current frame. As we have already proved, that the $N_f - q_{2,1}^{f,i}$ last reserved slots of u_i are each situated after a different reserved slot of u_{i-1}; this implies that u_i can send $N_f - q_{2,1}^{f,i}$ packets among the N_f it has received during the second frame. This means that node u_i has, at the beginning of the third frame, again $q_{2,1}^{f,i}$ pending packets. We can thus deduce that, from the second frame:

- Every intermediate node u_i 's queue length, is the same at the beginning of each frame.
- Every intermediate node u_i sends a packet at each reserved slot.

We propose the Algorithm 1 to compute, for any intermediate node u_i on the path flow, the number of packets it has at the beginning of the first slot of the second frame and so of every following frame. This algorithm relies on the fact that (1) the preceding node of u_{i-1} sends a packet at each of its reserved slot, (2) u_i forwards a packet at each of its reserved slot only if it has any pending packet in its queue, and (3) it has no pending packet at the beginning of the first frame.

5.2 Node's queue length at any slot

At the $(\Psi_j^{f,i-1})$th slot of the mth frame ($m > 1$), a node u_i's queue length equals to the number of pending packets it had at the beginning of the frame (i.e., $q_{2,1}^{f,i}$) plus the number of packets it has received since the beginning of the frame from u_{i-1} (i.e., j−1) minus the number (denoted $w(j)$) of packets it has already sent since the beginning of the interval.

Algorithm 1 Node u_i's queue length at the beginning of every frame $m > 1$

Require: $\Psi^{f,i}$ and $\Psi^{f,i-1}$
1: initialize $y = 1$ and $z = 1$
2: **while** $z \leq N_f$ **do**
3: **if** $\Psi_z^{f,i} > \Psi_y^{f,i-1}$ *and* $z \leq N_f$ **then**
4: $y++$ and $z++$ ▷ u_i forwards the packet received at the slot number $\Psi_y^{f,i-1}$ at the slot $\Psi_z^{f,i}$
5: **else**▷ u_i does not send any packet during the slot number $\Psi_z^{f,i}$ because it has no pending packets
6: $z++$
7: **end if**
8: **end while**
9: $q_{2,1}^{f,i} = N_f - y + 1$ ▷ number of received packets minus the number of sent packets
10: **return** $q_{2,1}^{f,i}$

It implies that an intermediate node u_i queue length, at the $(\Psi_j^{f,i-1})$th slot of the mth frame ($m > 1$), is equal to $j - 1 + q_{2,1}^{f,i} - w(j)$. Thus, at every frame (after the first one), a node's queue length at slot j is always the same:

$$\forall j \in [1; N] \; and \; \forall m > 1, q_{m,j}^{f,i} = q_{m+1,j}^{f,i} \tag{11}$$

The Algorithm 2 returns the queue length of an intermediate node u_i at the beginning of each reserved slot of its previous node u_{i-1} in the flow path. This algorithm relies on the fact that (1) node u_i sends a packet at each of its reserved slot, (2) u_i also sends a packet at each of its reserved slot, and (3) u_i has $q_{m,1}^{f,i}$ pending packet at the beginning of the frame.

Algorithm 2 Node u_i's queue length at the zth slot of any mth frame (with $m > 1$ and $z = \Psi_{m,j}^{f,i-1}$)

Require: $q_{m,1}^{f,i}$, j, $\Psi^{f,i}$ and $\Psi^{f,i-1}$
1: initialize $y = 1$
2: $q_{m,z}^{f,i} = q_{m,1}^{f,i} + j - 1$ ▷ u_i has received $j-1$ packets since the beginning of the frame and had $q_{m,1}^{f,i}$ pending packets since its beginning
3: **while** $y \leq N_f$ and $\Psi_{m,y}^{f,i} < z$ **do**
4: $q_{m,z}^{f,i}--$ ▷ u_i sends a packet at the slot number $\Psi_{m,y}^{f,i}$
5: $y++$
6: **end while**
7: **return** $q_{m,z}^{f,i}$

5.3 Packet's delay at a node

From the second frame, we have shown that every intermediate node sends a packet, at each of its reserved slot, and forwards a packet received at slot j, always at the same slot. If an intermediate node u_i on the flow path receives a packet, at the $(\Psi_j^{f,i-1})$th slot of the mth frame (with $\Psi_j^{f,i-1} = z$)

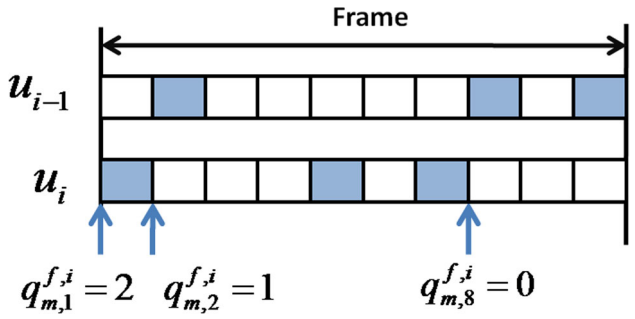


Fig. 5 Node u_i and u_{i-1}'s scheduling

while it has in its queue $q_{m,z}^{f,i}$ pending packets of f, it forwards it at the slot $\Psi_x^{f,i}$, such as:

$$x = \begin{cases} \phi_i^f(j) + q_{m,z}^{f,i} & \text{if } \phi_i^f(j) + q_{m,z}^{f,i} \\ & \text{is inferior or equal to } N_f \\ (\phi_i^f(j) + q_{m,z}^{f,i}) \% N_f & \text{else} \end{cases} \tag{12}$$

The function $\phi_i^f : [1, N_f] \rightarrow [1, N_f]$ returns, for an index $j \leq N_f$, the index of the first reserved slot of link u_i situated after the slot $\Psi_j^{f,i-1}$. To illustrate this formula, we propose the following example (see Fig. 5). The figure shows the scheduling of two intermediate nodes on the flow path f: nodes u_{i-1} and u_i. We can also observe that $N_f = 3$. From Algorithm 1, we can compute the queue length of u_i at the beginning of the second frame and at the beginning of every frame except the first one; we get $q_{m,1}^{f,i} = 2$ when $m > 1$. At the beginning of the first reserved slot of node u_{i-1} (the slot number 2), u_{i-1} starts sending a packet to node u_i, u_i has then in its queue one pending packet ($q_{m,2}^{f,i} = 1$) as it has sent a packet at its first reserved slot which is the first slot of the frame.

From Fig. 5, we can observe that the index of the first reserved slot of link u_i, which is situated after the first reserved slot of node u_{i-1}, is 2 (i.e., $\phi_i^f(1) = (2)$). Thus, according to Formula 12, as $\phi_i^f(1) + q_{m,2}^{f,i} = 3$ and $3 \leq N_f$, we can assert that u_i forwards the packet it has received, at the slot number 2, from node u_{i-1} at its third reserved slot, i.e., at the slot number $\Psi_3^{f,i} = 7$.

In the following, we denote by $\varphi_i^f : [1, N_f] \rightarrow [1, N_f]$, the function which returns for a given value j, the value of the index x, such as when e_{i-1} sends a packet at the $\Psi_j^{f,i-1}$ of the mth frame ($m \geq 2$), u_i forwards it at the slot $\Psi_x^{f,i}$ of the current or the next frame.

Thus, from the second frame, the delay of a packet at an intermediate node can take N_f different values according to the slot at which its arrives at that node. The delay (in slots) at an intermediate node u_i, for a packet arrived at the $(\Psi_j^{f,i-1})$th slot of the mth frame ($m \geq 2$), is:

$$d_i^f(j) = \begin{cases} \Psi_x^{f,i} - \Psi_j^{f,i-1} & \text{if the result is positive} \\ N - \Psi_j^{f,i-1} + \Psi_x^{f,i} & \text{else} \end{cases} \tag{13}$$

the value x is computed with Eq. 12.

5.4 Flow's delay

In the previous section, we have proposed a formula which computes the delay of any packet at any intermediate node u_i of any frame (except the first one). We denote $d_i^f(j)$, the delay of a packet at a node u_i when it receives the packet at the jth reserved slot of its preceding node on the flow path, this delay is computed with Formula 13. Thus, the packet at a node can take only N_f different values. Furthermore, as each node, from the second frame, always forwards a packet it receives at the jth slot at the same xth slot, then the delay of a packet along the path can also only take N_f different values. From the second frame, the delay of a packet depends on the slot at which the source (node u_1) sends it. If it sends it at the $(\Psi_j^{f,1})$th slots, the packet's delay (in slots) is:

$$d^f(j) = d_2^f(j) + d_3^f(\varphi_2^f(j)) + \cdots + d_n^f(\varphi_{n-1}^f(\ldots\varphi_2^f(j)) \tag{14}$$

As a reminder, we denote $\varphi_i^f : [1, N_f] \rightarrow [1, N_f]$, the function which returns for a given value j, the value of the subscript x, such as when u_{i-1} sends a packet at the $\Psi_j^{f,i-1}$ of the mth frame ($m > 1$), u_i forwards it at the slot number $\Psi_x^{f,i}$ of the current or the next frame. In Eq. 14, as the delay of a packet at the source (node u_1) and the destination (node u_{n+1}) is assumed to be null, there is no delay computed at these nodes. Finally, the maximum delay of a flow is the maximum delay that a packet can reach:

$$d^f = \max_{i \in [1, N_f]} (d^f(i)) \tag{15}$$

6 A modified branch and bound algorithm

In this section, we propose an algorithm, named ACL2S, which solves the admission control with link scheduling problem. ACL2S is an iterative algorithm which integrates a modified version of the Dakin's branch and bound (B&B) method [48] (MB&BA). In this section, we first describe ACL2S's iterative algorithm. Then, we introduce the MB&BA (i.e., our modified version of the Dakin's B&B method).

6.1 ACL2S's iterative algorithm

ACL2S is an iterative algorithm. At each iteration, in a first step, it solves the BL2P problem thanks to our MB&BA. Our MB&BA returns a scheduling for the flow with a low delay. In a second step, our algorithm verifies whether the delay of the returned scheduling guarantees that the flow's delay is inferior or equal to the required maximum delay. If it is the case, the scheduling is accepted and the algorithm stops; otherwise, the algorithm returns to the first step after having modified the BL2P so that the previous obtained scheduling are no longer solutions of the problem. To modify the BL2P, new linear constraints are added to the problem; in other words, a binary cut is performed on the initial problem. The algorithm returns then to the first step. The algorithm stops when (1) one scheduling respects the required delay, or (2) the number of iterations is superior to a certain threshold β, or (3) when the MB&BA does not find any longer a solution to the BL2P. The ACL2S is summarized in Fig. 6.

6.2 MB&BA: a modified version of the B&B algorithm

The Dakin's B&B algorithm enables to solve linear programming problems and, among others, BL2P. This algorithm systematically enumerates all feasible solutions of the problem where large subset of candidates are discarded by using lower and upper estimation bounds of the quantity being optimized which is, in our case, the quantity obtained with Eq. 10. The B&B method integrates an algorithm which solves the linear programming problem. In the following, we have chosen the Simplex method as the underlying method to solve the linear programming problem as it is a simple and yet efficient solution [49]. To solve a BL2P, which for example has to return a matrix S^f of size $|V|*|N_s|$ of binary elements, the Dakin's B&B method first initializes $z*$ the lower bound of the objective function. $z*$ is equal to the first integer superior or equal to the objective value of the solution obtained for the relaxed BL2P with the Simplex method. A relaxed BL2P does not consider the binary constraint of the solution. The Dakin's method also initializes the upper bound of the solution z' at $+\infty$. Then, it creates a first node which is the root of a tree. This node is activated and associated with (1) the original BL2P, (2) the solution S^f obtained with the simplex method over the relaxed BL2P, and (3) the objective value z' of the solution S^f. Then, the algorithm performs iteratively the following steps.

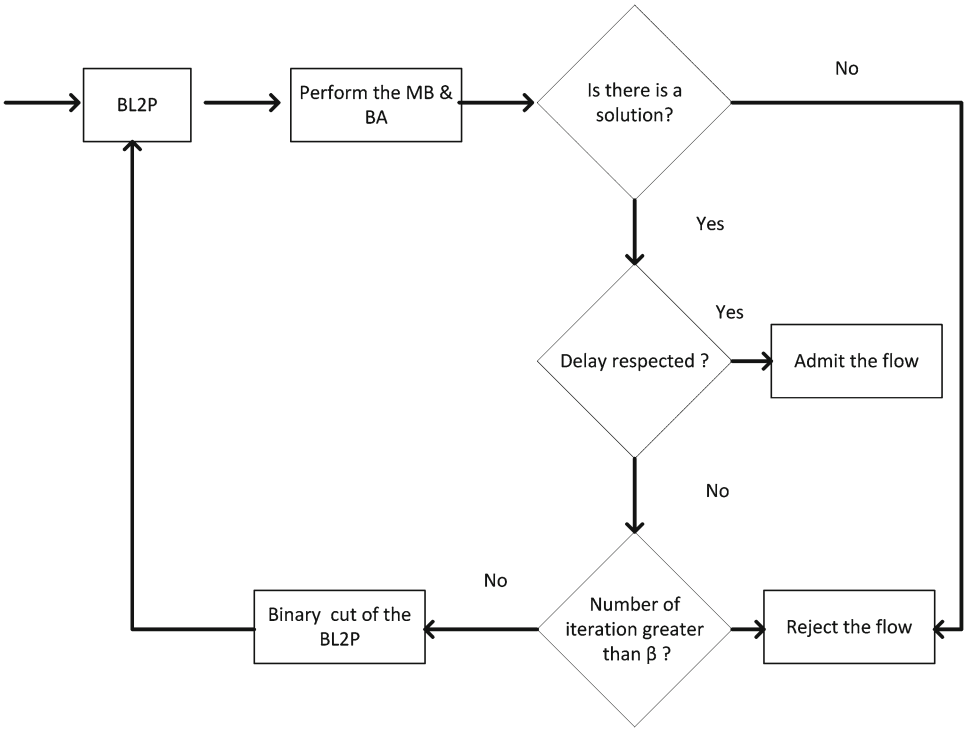
1. It chooses an activated node and creates two child nodes.
2. Each child is activated and is associated with the BL2P of its parent node at which a constrain is added. The added constraint is chosen as follows. An element s_{ij}^f of its parent solution S^f, which does not have a binary value, is arbitrarily chosen. The first child node's added constraint is $s_{ij}^f = 0$, whereas the second child node's added constraint is $s_{ij}^f = 1$.
3. The Simplex algorithm is performed for each relaxed BL2P of the two child nodes. If the Simplex returns no solution for one child node, this latter is deactivated.
4. The solution returned by the Simplex and its associated objective value z is assigned to each child node.
5. If every element of the solution of one child node is binary and if $z < z'$, then z' takes the value z. If a node has a binary solution then it is deactivated. Finally, the algorithm returns to step 1.

B&B stops either when one node has a binary solution whose objective value is equal to $z*$ (which is a solution of the BL2P), or when there is no more activated node. If there is no more activated node and if $z' = +\infty$, there is no solution to the problem. If there is no more activated node and if

Algorithm 3 MB&BA: Modified Branch and Bound Algorithm

Require: BL2P
1: $z' = +\infty$
2: Set the first active node u_0 of the tree, assign it the BL2P, its solution matrix S_0^f and its objective value z_0
3: z^*= first integer superior or equal to z_0
4: **while** there are still active nodes and no node has a binary matrix which objective value is z^* **do**
5: Select an active node u_x in the tree, its associated matrix is S_x^f
6: Create two child nodes to u_x
7: Select a variable s_{ij} such as e_i is the first link of the flow path which number n of reserved slots is inferior to N_f and which every preceding link of the flow path has more reserved slots.
8: If $e_i \neq l_1$, find the slot situated after the $n+1$th reserved slots of links l_{z-1} which is not binary for link e_i and initialize j to this slot number.
9: If $e_i = l_1$, then choose randomly j such as s_{ij} is not a binary element in S_x^f . If it is not possible to find such an element, return to line 4 and deactivate the two child nodes.
10: Associate, to each child node, the BL2P of u_x with an added constraint; $s_{ij}^f = 1$ for the first node and $s_{ij}^f = 0$ for the second
11: **for** each child u_y of u_x **do**
12: Solve its relaxed BL2P
13: **if** its relaxed BL2P has a solution **then**
14: Assign it its objective value z_y and its solution matrix S_y^f
15: **if** $z_y = z^*$ and S_y^f binary **then**
return $S^f = S_y^f$
16: **else if** $z_y > z'$ **then**
17: Deactivate node u_y
18: **else if** $z_y < z'$ and S_y^f a binary matrix **then**
19: $z' = z_y$
20: **end if**
21: **end if**
22: **end for**
23: **end while**
24: **if** $z' == +\infty$ **then return** $S^f = null$
25: **else return** binary matrix S^f
26: **end if**

Fig. 6 ACL2S



$z' \neq +\infty$, then the BL2P's solution is the binary solution which objective value is equal to z'.

We enhanced the algorithm so that the constraint added to each child node guarantees that the solution, returned by the algorithm, offers a low delay for the flow. To reach this goal, the proposed algorithm selects, for each intermediate node on the flow path, slots which are as close as possible from the scheduled slots of the previous node on the flow path and situated after these latter. Thus, a packet does not wait long at a node. As a reminder, the path of a flow f is denoted $p_f = (l_1, l_2 \ldots, l_n)$. The MB&BA chooses the element s_{ij}^f at the step 2 such as:

- it is not a binary element of the parent matrix S^f,
- the index i of s_{ij}^f refers to a link $e_i \in p_f$ ($e_i = l_z$) and e_i must be the first link over the flow path which has the less reserved slots and its number n of reserved slots must be inferior to N_f,
- if e_i is l_1, then the value j of the index s_{ij}^f is randomly chosen; otherwise, j is equal to the first slot situated after the $n+1$th reserved slot of link l_{z-1} for which s_{ij}^f is not binary at the parent matrix S^f.

If no element s_{ij}^f can be found, the algorithm deactivates the two current child nodes and returns to step 2. The matrix S^f returned by the algorithm represents a scheduling for the flow. This scheduling guarantees a low delay while respecting the bandwidth required. The Algorithm 3 summarizes our MB&BA. However, even if this scheduling guarantees a low delay, it does not guarantee that this delay is inferior or equal to the maximum delay that the flow requires, so that the MB&BA must be integrated in the ACL2S. Finally, the ACL2S algorithm admits a flow if it can return a scheduling which respects the flow constraints in terms of bandwidth and delay; otherwise, the flow is rejected.

7 Admission control with link scheduling

In our solution, the gateway decides the scheduling of every flow accepted in the WMN. It stores active flows scheduling using the scheduling matrix denoted S and described in Eq. 2 of Sect. 3. Therefore it has a global state information on the network and can decide whether it accepts or rejects a new flow. Every node in the network stores only a local state information on the network describing the scheduling of its links. Therefore, it only knows the slots during which it can activate a link to send packets for a specific flow.

Our solution relies on a reactive routing protocol like the Ad hoc On-Demand Distance Vector (AODV) Routing [50] and takes advantage of the two main messages of any reactive routing protocol; the route request (RREQ) and the route response (RREP) packets. As in the famous Resource Reservation Protocol-Traffic Engineering (RSVP-TE) [51], our solution reserves resources across a network for a flow by sending two packets; a modified RREQ and RREP. We have modified the RREQ sent by the source to the destination, in order that it contains the flow's requirements, like in the

RSVP-TEs Path message. Furthermore, we have modified the RREP, sent by the destination to the source, so that it reserves enough resources for the flow along the path taken by the RREQ, like in the Resv message of the RSVP-TE. The admission control process takes place in three steps; (1) the admission control request, (2) the admission control, and (3) the admission control reply.

7.1 Admission control request

The node, which receives a request to admit a flow, broadcasts a modified RREQ where it indicates, the minimum bandwidth and the maximum delay required by the flow. Every node, which receives the packet, checks whether it has not already received it and if its time to live has not expired. If this fails, then the node drops the packet; otherwise, it adds to it its ID before re-broadcasting it. This scheme goes on till the RREQ reaches a gateway.

7.2 Admission control

Every gateway knows the matrix scheduling of the network S, the receivers matrix L^r, the senders' matrix L^s, the power of the network's matrix P and every matrix of flow scheduling. All gateways of the network exchange their matrices so that they all have up-to-date matrices. When a gateway does not receive any packet of a flow, since a time t, it deletes its scheduling matrix and its entry in the network scheduling matrix and then informs the other gateways of the network. When a gateway receives a RREQ, it extracts from this packet the bandwidth and the maximum delay required by the flow. It then launches the ACL2S in order to find a schedule for the flow along the path followed by the RREQ. If a schedule is obtained, then it ignores the following RREQs it receives for the flow. It then unicasts a packet, a modified RREP, along the selected path where it indicates the scheduling of the new accepted flow.

7.3 Admission control reply

Upon receiving a RREP, a node extracts from the packet its scheduling for the flow. When the source node receives the RREP, it starts sending the packets of the flow. If the source, after a certain time, does not receive any RREP, then the source rejects the flow which is then not sent on the network. If a node does not receive, for a time t, any packet of the flow, then it deletes the scheduling of the flow. t's value is proportional to the rate required by the flow and to the packets' payload. For example, if a flow requires a rate of 96 kbit/s and if the packets payload is 1000 bytes, then a node on the flow path should receive in average one packet every 84ms. For safety reason, the value of t is fixed as ten times the average time between each packet, so that, the flow's schedule is still active after a short and temporary failure. Thus, in the example presented below, the value of t is 840 ms.

To summarize, when the WMN gateway receives a request RREQ to admit a new flow, it performs the ACLS algorithm. The RREQ packet specifies the flow requirements in the form of a triplet. This triplet defines respectively the number of slots per frame, the path and the minimum delay required for the flow. The gateway uses the ACLS algorithm to decide whether it can fulfill the flow requirements and outputs a scheduling for the flow. It sends a RREP message along the reverse path of the RREQ packet specifying whether the flow is accepted or not. If the flow is accepted, the RREQ packet specifies the flow scheduling. Every node along the path receiving RREP learns its scheduling for the flow. When the flow starts every node along the path sends the flows packets at the slots reserved for that flow.

8 Performance evaluation

To validate our model, we have implemented it on the event-driven simulator ns2 [52]. We have performed important changes on MAC modules of ns2 and more precisely on the Mac802.11Ext DropTail, and network classes. In order to solve the ACLS algorithm, the network module uses the GNU Linear Programming Kit [53], which is a package for solving large-scale linear programming. We have performed a cross layer between the queue and network modules so that the queue module is aware of the scheduling of a node. Thus, the queue module sends a packet of data to the MAC module at the beginning of the slot scheduled for the packet's sending. Furthermore, the backoff procedure of the MAC module has been modified for every packet of data, so that it sends a packet of data directly after having received it from the queue module. We have chosen to compare our solution ACL2S with BRAWN admission control model [54]. BRAWN is based on the computation of the available bandwidth by each node in the network. BRAWN's implementation on ns2 simulator is freely available on the Internet [55]. We choose to compare our solution to BRAWN for two reasons:

- as BRAWN is a simple and yet mature and effective admission control solution.
- in order to show the performance difference between our admission control which relies on a link scheduling scheme and an efficient admission control which relies on CSMA/CA.

We compared these solutions in terms of delay, throughput and aggregated throughput according to two network topologies: linear and cross (see Figs. 7, 8), where nodes are separated from each other by 90 meters.

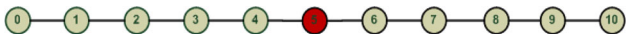


Fig. 7 A linear topology made up of 11 nodes; one of which (i.e., node 5) is a gateway

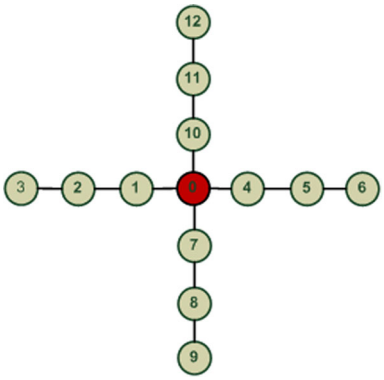


Fig. 8 A cross topology made up of 13 nodes; one of which (i.e., node 0) is a gateway

Table 1 Simulation parameters

Layer	Parameters	Values
Physical layer	One antenna per node	
	Carrier sensing layer	6.31e−9 mW
	Frequency	2.4 GHz
	Transmission power	100 mW
	Rate	54 Mbit/s
MAC layer	T_{slot}	250 µs
	RTS/CTS mechanism	Off
	N for the linear topology	164
	N for the cross topology	172
Transport layer	UDP size packet	1000 bytes

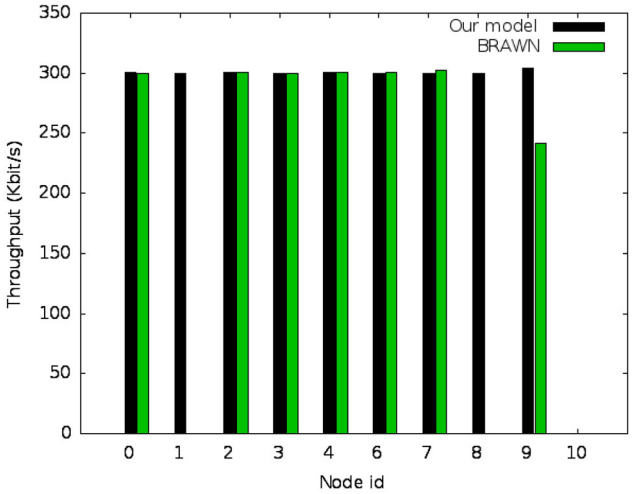


Fig. 9 Nodes' throughput in a WMN in a linear topology

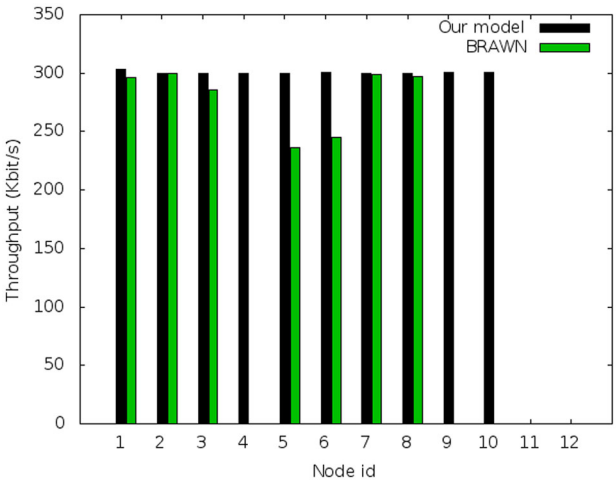


Fig. 10 Nodes' throughput in a WMN in a cross topology

The parameters used for the simulations are quite usual and are presented in Table 1. However, we introduced two new parameters: N represents the number of slots per frame and T_{slot} is the length of a slot. T_{slot} is equal to the time needed for a node to send a packet of data. N has been chosen in order to maximize the performance of our solution; its value is different according to the network topology.

For both topologies, each node sends a flow of 300 kbit/s and requires a delay of maximum 150 ms. The nodes send a request for their flows admission at one second of interval from each other. Every node i sends a request for its flow at the end of the $i + 1$th second unless it is a gateway. Figures 9 and 10 show the throughput reached by each node of the network (except the gateway) in a WMN with the linear and the cross topology. We can observe that our solution, with the linear topology, admits 9 flows out of 10 requests of admission, whereas BRAWN admits 7 flows. ACLS, with the cross topology, admits 10 flows out of 12, whereas BRAWN admits 7 flows. Indeed, BRAWN admits less flow (it rejects flows 4, 9, 10, 11, 12 for the cross topology and 1 and 10 for the linear topology) than our solution (it rejects flows 11 and 12 for the cross topology and flow 10 for the linear) as it has less available bandwidth due to the use of the backoff procedure which can be quite long and the retransmission of many data packets due to collisions. Thus, for the cross topology, we can assume that flows 9, 10, 11 and 12 were rejected because there was no more available bandwidth on the network when the requests to admit these flows were sent. Indeed, the bandwidth has then been reserved for the flows 1, 2, 3, 5, 6, 7 and 8. We can also assume that flow 4 has not been admitted due to the loss of its admission request.

Furthermore, we can notice that our solution, ACLS, respects every flow's expected throughput while it is not always the case for BRAWN. For example, with the linear topology, the flow sent by node 9 does not reach its expected 300 kbit/s. We can also observe that, with the cross

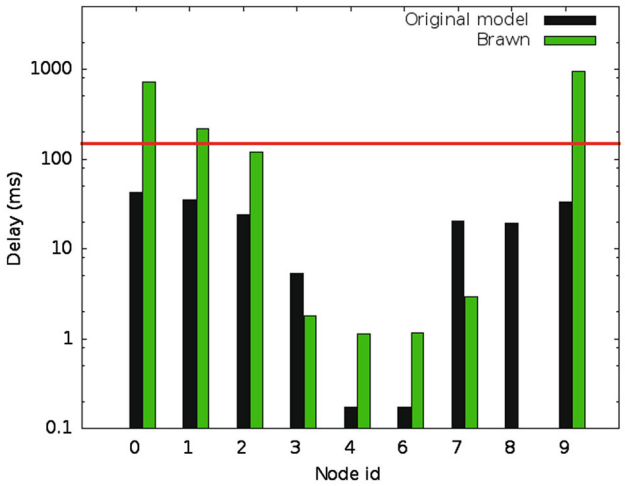


Fig. 11 Flows' delay in a WMN with a linear topology

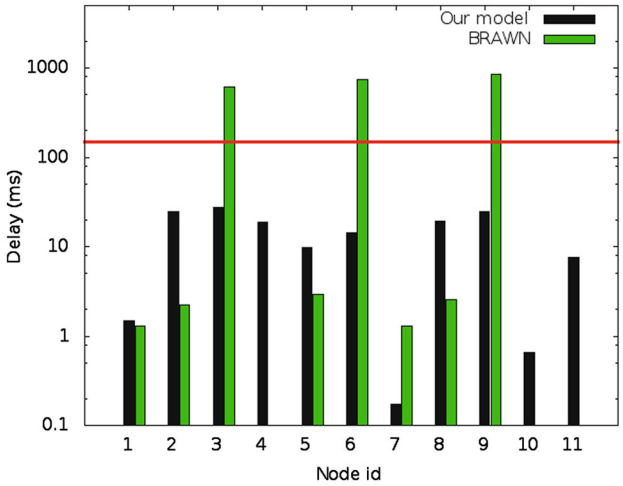


Fig. 12 Flows' delay in a WMN with a cross topology

topology, the flows sent by the nodes 5 and 6 do not reach their expected throughput .We can assume that they do not reach their required throughput due to collisions (BRAWN is not collision free) and because the nodes, with BRAWN, may overestimate their available bandwidth before accepting these flows. Furthermore, we can notice on the cross topology than flows 3, 8 and 9 have not totally reached their bandwidth, it may be due to collisions or because some of their packets are still in pending at intermediate nodes at the end of the simulation. Our solution reaches better performance in terms of both throughput and number of admitted flows. Indeed, our link scheduling scheme offers efficient use of bandwidth for each node, thus the network can admit more flows. Furthermore, the proposed link scheduling scheme enables to avoid collisions and unfairness compared to CSMA/CA and can better guarantee the throughput of each admitted flow. The problem of unfairness induced by CSMA/CA in WMNs has already been observed in existing papers [7,10,56,57].

Figures 11 and 12 show the mean delay of each flow with the linear and cross topologies. We can observe that ACLS, in both topologies, always guarantees that the flows' delay is inferior or equal to the maximum delay required by the flow. The maximum delay is represented by the red horizontal line. In contrary, BRAWN does not respect the delay required by each flow; when the source of the flow is far away from the gateway, its delay is important. We can observe in both topologies that, when the source of the flow is at three hops or more from the gateway, the maximum delay of the flow is no longer respected except for the node 2 in the linear topology. However, the node 2's delay in the linear topology is quite high. The good performance in terms of delay of ACLS can be explained by the fact that our model admits a flow if it can respect both its delay and bandwidth, whereas BRAWN only aims at respecting flows' throughput. Furthermore, BRAWN suffers from the unfairness of CSMA/CA.

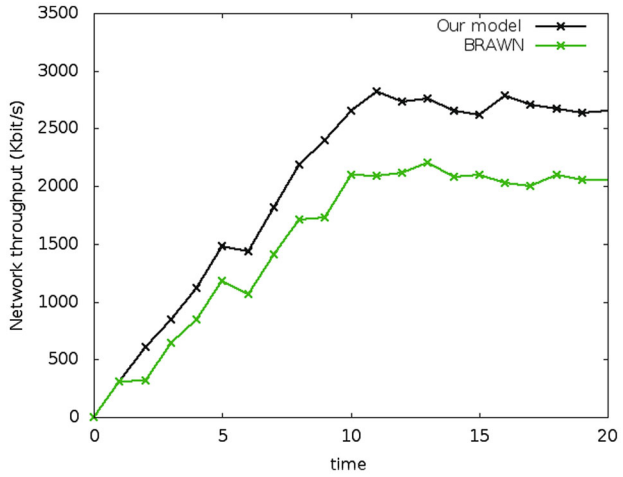


Fig. 13 Aggregated throughput in a WMN with a linear topology

Figures 13 and 14 show the aggregated throughput with, respectively, the linear and the cross topology. Our solution shows better performance than BRAWN in both topologies. We can also notice that, in both solutions and topologies, the aggregated throughput increases with the number of accepted flows till they cannot admit any more flow; then, the throughput stays quite stable. In both topologies, the difference between the aggregated throughput of each solution increases with the network's load. Indeed, the more the network's load is important, the more CSMA/CA may lead to collisions and large backoff timer. Furthermore, some studies have shown that scheduling could improve the throughput of the network [27,46] and this fact is also illustrated by these results.

Finally, thanks to these results, we observe that ACLS reaches its goal of admitting more flows than admission controls based on CSMA/CA. Furthermore, it has also pointed out that ACLS respects the bandwidth and delay of the admitted flows, which may not always be the case of solutions

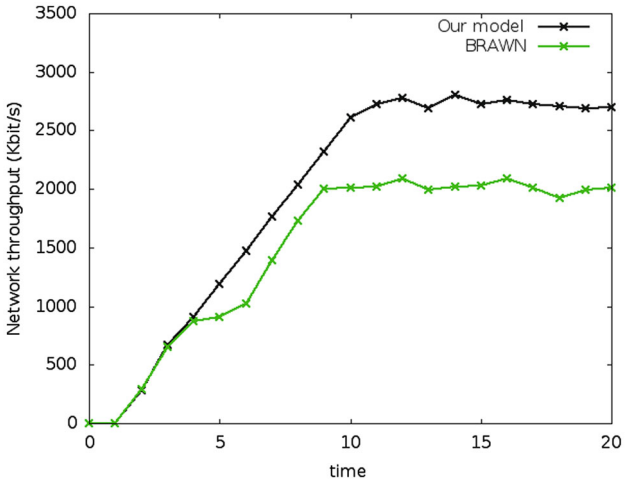


Fig. 14 Aggregated throughput in a WMN with a cross topology

based on a competition access. Indeed, the schemes which rely on CSMA/CA may suffer from collisions and unfairness between nodes.

9 Conclusions

We proposed a new admission control based on link scheduling, named ACLS. It is, to the best of our knowledge, one of the first works which integrates two separate concepts: admission control and link scheduling. The idea is to make the network scheduling evolves each time a new flow is admitted in the network or stops being active. An original method is proposed to compute the maximum delay that can reach a flow according to its scheduling. We also modeled the problem of admission control with link scheduling based on bandwidth as a binary linear programming problem. We introduced an iterative algorithm based on the Dakin's B&B method which aims at computing a scheduling for a new flow which respects both its delay and bandwidth. Furthermore, the proposed iterative algorithm is integrated in a complete admission control scheme. The simulation results showed that ACLS model has better performance than the reference solution BRAWN. We believe that the proposed solution can be easily implemented in real 802.11 WMNs. Indeed, (1) the proposed solution is fully transparent to users and, (2) the IEEE 802.11 amendment for mesh networking offers techniques that allow nodes' synchronization and time division multiple access. As future work, we plan to consider different flow types in the proposed model in order to provide QoS in real networks.

References

1. Khatoun, R., & Zeadally, S. (2016). Smart cities: Concepts, architectures, research opportunities. *Communications of the ACM*, *59*, 46–57.
2. Doraghinejad, M., Nezamabadi-pour, H., & Mahani, A. (2014). Channel assignment in multi-radio wireless mesh networks using an improved gravitational search algorithm. *Journal of Network and Computer Applications*, *38*, 163–171.
3. Lenzini, L., Mingozzi, E., & Vallati, C. (2010). A distributed delay-balancing slot allocation algorithm for 802.11s mesh coordinated channel access under dynamic traffic conditions. In *MASS* (pp. 432–441). IEEE.
4. Krasilov, A., Lyakhov, A., & Safonov, A. (2011). 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* (pp. 752–757). Washington, DC: IEEE Computer Society.
5. Lee, J., Yoon, H., & Yeom, I. (2010). Distributed fair scheduling for wireless mesh networks using ieee 802.11. *IEEE Transactions on Vehicular Technology*, *59*, 4467–4475.
6. Luo, L., Gruteser, M., Liu, H., Raychaudhuri, D., Huang, K., & Chen, S. (2006). A qos routing and admission control scheme for 802.11 ad hoc networks. In *DIWANS '06*, ACM.
7. Raniwala, A., De, P., Sharma, S., Krishnan, R., & Chiueh T.C. (2007). End-to-end flow fairness over ieee 802.11-based wireless mesh networks. In *INFOCOM*, (pp. 2361–2365). IEEE.
8. IEEE standard for information technology—Specific requirements part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications amendment 10: Mesh networking. *IEEE Std 802.11s-2011* (2011).
9. Hiertz, G., Denteneer, D., Max, S., Taori, R., Cardona, J., Berlemann, L., et al. (2010). Ieee 802.11s: The wlan mesh standard. *IEEE Wireless Communications*, *17*, 104–111.
10. Khoukhi, L., Badis, H., Merghem-Boulahia, L., & Esseghir, M. (2013). Admission control in wireless ad hoc networks: A survey. *EURASIP Journal on Wireless Communications and Networking*, *2013*, 1–13.
11. Chakraborty, S., & Nandi, S. (2015). Distributed service level flow control and fairness in wireless mesh networks. *IEEE Transactions on Mobile Computing*, *14*, 2229–2243.
12. Jun, J., & Sichitiu, M. L. (2003). The nominal capacity of wireless mesh networks. *Wireless Communication*, *10*(5), 8–14.
13. Dhurandher, S., Woungang, I., Kumar, K., Joshi, M., & Verma, M. (2012). A rate adaptive admission control protocol for multimedia wireless mesh networks. In *2012 IEEE 26th international conference on advanced information networking and applications (AINA)* (pp. 38–43).
14. Rezgui, J., Hafid, A., & Gendreau, M. (2010). Distributed admission control in wireless mesh networks: Models, algorithms, and evaluation. *IEEE Transactions on Vehicular Technology*, *59*, 1459–1473.
15. Dhurandher, S.K., Woungang, I., Obaidat, M.S, Kumar, K., Joshi, M., & Verma, M. (2014). A distributed adaptive admission control scheme for multimedia wireless mesh networks. *IEEE Systems Journal, 9*(2), 595–604. doi:10.1109/JSYST.2013.2296336.
16. Dhurandher, S. K., Woungang, I., Obaidat, M. S., Kumar, K., Joshi, M., & Verma, M. (2015). A distributed adaptive admission control scheme for multimedia wireless mesh networks. *IEEE Systems Journal*, *9*, 595–604.
17. Khoukhi, L., & Cherkaoui, S. (2005). A quality of service approach based on neural networks for mobile ad hoc networks. In *Second IFIP international conference on wireless and optical communications networks, 2005. WOCN 2005* (pp. 295–301).
18. Dromard, J., Khoukhi, L., & Khatoun, R. (2013). An efficient admission control model based on dynamic link scheduling in wireless mesh networks. *EURASIP Journal on Wireless Communications and Networking*, *2013*(1), 1–18.
19. Dromard, J., Khatoun, R., & Khoukhi, L. (2013). A watchdog extension scheme considering packet loss for a reputation system

in wireless mesh network. In *2013 20th international conference on telecommunications (ICT)* (pp. 1–5).
20. Ivesic, K., Skorin-Kapov, L., & Matijasevic, M. (2014). Cross-layer qoe-driven admission control and resource allocation for adaptive multimedia services in LTE. *Journal of Network and Computer Applications*, *46*, 336–351.
21. Zhao, Z., Mu, J., & Guan, H. (2010). Characterizing the end-to-end throughput in wmn using multiple directional antennas. In *2010 IEEE international conference on wireless communications, networking and information security (WCNIS)*.
22. Ding, Y., Pongaliur, K., & Xiao, L. (2013). Channel allocation and routing in hybrid multichannel multiradio wireless mesh networks. *IEEE Transactions on Mobile Computing*, *12*, 206–218.
23. Marina, M. K., Das, S. R., & Subramanian, A. P. (2010). A topology control approach for utilizing multiple channels in multi-radio wireless mesh networks. *Computer Networks*, *54*, 241–256.
24. Akyildiz, I. F., Wang, X., & Wang, W. (2005). Wireless mesh networks: A survey. *Computer Networks and ISDN Systems*, *47*, 445–487.
25. Ian Akyildiz, X. W. (2009). *Wireless mesh networks*. Chichester: Wiley.
26. Nabet, A., Khatoun, R., Khoukhi, L., Dromard, J., & Gati, D. (2011). Towards secure route discovery protocol in manet. In *Global information infrastructure symposium—GIIS 2011* (pp. 1–8).
27. Brar, G., Blough, D. M., & Santi, P. (2006). 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* (pp. 2–13). New York, NY: ACM.
28. Gass, S. I. (1984). *Linear programming: methods and applications* (5th ed.). New York, NY: McGraw-Hill, Inc.
29. Gore, A., & Karandikar, A. (2011). Link scheduling algorithms for wireless mesh networks. *IEEE Communications Surveys Tutorials*, *13*, 258–273.
30. Sharma, G., Mazumdar, R. R., & Shroff, N. B. (2006). On the complexity of scheduling in wireless networks. In *Proceedings of the 12th annual international conference on mobile computing and networking* (pp. 227–238). New York, NY: ACM.
31. Vieira, F. R., de Rezende, J. F., Barbosa, V. C., & Fdida, S. (2012). Scheduling links for heavy traffic on interfering routes in wireless mesh networks. *Computer Networks*, *56*, 1584–1598.
32. Xu, K., Gerla, M., & Bae, S. (2002). How effective is the ieee 802.11 rts/cts handshake in ad hoc networks. In *Global telecommunications conference, 2002. GLOBECOM '02. IEEE* (vol. 1, pp. 72–76).
33. Jain, K., Padhye, J., Padmanabhan, V. N., & Qiu, L. (2003). Impact of interference on multi-hop wireless network performance. In *Proceedings of the 9th annual international conference on mobile computing and networking* (pp. 66–80). ACM.
34. Gore, A., Karandikar, A., & Jagabathula, S. (2007). On high spatial reuse link scheduling in stdma wireless ad hoc networks. In *Global telecommunications conference, 2007. GLOBECOM '07. IEEE* (pp. 736–741).
35. Blough, D. M., Resta, G., & Santi, P. (2010). Approximation algorithms for wireless link scheduling with sinr-based interference. *IEEE/ACM Transactions on networking*, *18*, 1701–1712.
36. Cappanera, P., Lenzini, L., Alessandro, L., Stea, G., & Vaglini, G. (2013). Optimal joint routing and link scheduling for real-time traffic in tdma wireless mesh networks. *Computer Networks*, *57*, 2301–2312.
37. Rhee, I., Warrier, A., & Min, J. (2006). Drand: Distributed randomized tdma scheduling for wireless ad hoc networks. In *Proceedings of the Seventh ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2006. Florence, Italy. (pp. 190–201). ACM New York, NY, USA.
38. Iyer, A., Rosenberg, C., & Karnik, A. (2009). What is the right model for wireless channel interference? *IEEE Transactions on Wireless Communications*, *8*, 2662–2671.
39. Hanzo, L., & Tafazolli, R. (2009). Admission control schemes for 802.11-based multi-hop mobile ad hoc networks: a survey. *IEEE Communications Surveys & Tutorials*, *11*, 78–108.
40. Yang, Y., & Kravets, R. (2005). Contention-aware admission control for ad hoc networks. *IEEE Transactions on Mobile Computing*, *4*(4), 363–377.
41. Shen, Q., Fang, X., Li, P., & Fang, Y. (2009). Admission control based on available bandwidth estimation for wireless mesh networks. *IEEE Transactions on Vehicular Technology*, *58*, 2519–2528.
42. Bai, Y.-B., Zhu, X., Shao, X., & Yang, W.-T. (2012). Fast: Fuzzy decision-based resource admission control mechanism for manets. *Mobile Networks and Applications*, *17*, 758–770.
43. Shila, D. M., & Anjali, T. (2010). An interference-aware admission control design for wireless mesh networks. *EURASIP Journal on Wireless Communications and Networking*, *201*, 1.
44. Dromard, J., Khoukhi, L., & Khatoun, R. (2012). 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 (pp. 399–412), Berlin: Springer.
45. Yu, X., Navaratnam, P., Moessner, K., & Cruickshank, H. (2015). Distributed resource reservation in hybrid mac with admission control for wireless mesh networks. *IEEE Transactions on Vehicular Technology*, *64*, 5891–5903.
46. Gore, A., & Karandikar, A. (2011). Link scheduling algorithms for wireless mesh networks. *IEEE Communications Surveys Tutorials*, *13*, 258–273.
47. Jabbari, B., & Babaei, A. (2012). Interference and its impact on system capacity. In J.D. Gibson (Ed.) *Mobile communications handbook* (3rd ed., pp. 369–388). ACM New York, NY, USA.
48. Dakin, R. (1965). A tree search algorithm for mixed integer programming problems. *The Computer Journal*, *8*, 250–255.
49. Bertsimas, D., & Tsitsiklis, J. (1997). *Introduction to linear optimization*. Belmont: Athena Scientific.
50. Perkins, C., Belding-Royer, E., & Das, S. (2003). Ad hoc on-demand distance vector (aodv) routing, IETF RFC 3561, RFC Editor, United States, July 2003.
51. Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., & Swallow, G. (2014). RFC 3209 (Proposed Standard). Accessed July 19, 2014.
52. The Network Simulator NS-2. http://www.isi.edu/nsnam/ns/. Accessed July 19, 2014.
53. GLPK (GNU linear programming kit). http://www.gnu.org/software/glpk. Accessed December 11, 2014.
54. Guimares, R., Cerd, L., Barcel, J. M., Garca, J., Voorhaen, M., & Blondia, C. (2009). Quality of service through bandwidth reservation on multirate ad hoc wireless networks. *Ad Hoc Networks*, *7*(2), 388–400.
55. Voorhaen, M. (2014). Performance analysis of telecommunication systems. http://www.pats.ua.ac.be/software/brawn/. Accessed July 19, 2014.
56. Aoun, B., & Boutaba, R. (2006). Max–min fair capacity of wireless mesh networks. In *2006 IEEE international conference on mobile adhoc and sensor systems (MASS)*.
57. Masri, A. E., Sardouk, A., Khoukhi, L., Hafid, A., & Gaiti, D. (2014). Neighborhood-based and overhead-free congestion control for ieee 802.11 wireless mesh networks. *IEEE Transactions on Wireless Communications*, *14*, 2229–2243.

Dr. J. Dromard has received her engineering degree in 2010 in Information and Telecommunication Systems from the University of Technology of Troyes (UTT). In 2013, she got a doctor's degree in Network, Knowledge and Organization from UTT for her thesis entitled "Towards a secure admission control in a wireless mesh network"s. She is currently in post-Doctoral position in LAAS-CNRS. Her research interests are in the areas of unsupervised network anomaly detection, unsupervised machine learning techniques, big data, mesh networks and admission control.

Dr. L. Khoukhi received the Ph.D. degree in electrical and computer engineering from the University of Sherbrooke, Sherbrooke, QC, Canada, in 2006. In 2008, he was a Researcher with the Department of Computer Science and Operations Research, University of Montreal. Since 2009, he is an Assistant Professor with the University of Technology of Troyes, France. He has more than 70 publications in reputable journals and conferences. His research interests include vehicular networks, Machine-to-Machine Communications, resources management, attacks detection, and communication protocols.

Dr. R. Khatoun received his M. Sc in Computer Engineering and the Ph.D from the University of Technology of Troyes (UTT) in France in 2004 and 2008. He is currently associate professor at Telecom Paris Tech. His research interests include DDoS attacks detection and defense, intrusion detection system, mobile ad hoc networks security and computer security infrastructure.

Dr. Y. Begriche received the doctorat of appied mathematics from the university Paris V in France in 1988. Since 2005 he teaches at the Paris-Tech Institute (Paris) and participates in research activities in the department "informatique et réseaux" (computer and networks) (INFRES) of the Paris-Tech Institute. His area of research is the application of mathematics in the field of science and technology information, mobile, ad hoc and vehicular networks.