Conflict graph-based Markovian model to estimate throughput in unsaturated IEEE 802.11 networks

Marija Stojanova, Thomas Begin, Anthony Busson

Conflict graph-based Markovian model to estimate throughput in unsaturated IEEE 802.11 networks

Marija STOJANOVA marija.stojanova@ens-lyon.fr
Thomas BEGIN thomas.begin@ens-lyon.fr
Anthony BUSSON anthony.busson@ens-lyon.fr
Université Lyon 1, ENS Lyon, Inria, CNRS, UMR 5668 - Lyon, France

***Abstract*—WLANs (Wireless Local Area Networks) have become ubiquitous in our everyday life, and are mostly based on IEEE 802.11 standards. In this paper, we consider the performance evaluation of an arbitrary-topology unsaturated network based on the IEEE 802.11 DCF. We present a conflict graph-based modeling approach to discover the attainable throughput of each node. Our model consists of a single Markov chain which aims at describing, at a high-level of abstraction, the current state of the entire wireless network. Owing to its low complexity, our approach is simple to implement, can cope with medium sized networks, and its execution speed is fast. We validate its accuracy against a discrete-event simulator. Results show that our approach is typically accurate, with associated relative errors generally less than 15%, and that it captures complex phenomena such as node starvation. We investigate two potential applications of our proposed approach in which, starting with a given network, we improve its performance in terms of overall throughput or fairness by throttling the throughput demand of a node, or by turning a node off altogether.**

I. Introduction

WLANs (Wireless Local Area Networks) have become ubiquitous in our everyday life. Internet service providers increasingly employ WLANs to connect users to their network. WLANs are also widespread in airports, train stations, shopping malls, and cities that offer wireless Internet access to their customers. Most WLANs are based on IEEE 802.11 standards. Despite continuous improvements of 802.11 and amendments in its subsequent versions, 802.11-based networks typically feature transmission ranges and data rates that may both be deemed small given the soaring amount of end-users.

These concerns have been largely accommodated by network densification, which refers to the deployment of more APs (Access Points) per unit of area. By doing so, WLANs have improved their spatial reuse of the spectrum, as well as their their transmission rate. However, the management of what used to be a single AP with a few users has become much more complex. WLANs with several APs and tens of users are prone to misconfiguration and poor coordination between APs. Possible effects include inefficient use of the WLAN radio resources, poor overall WLAN performance, and unfairness among users.

Nowadays, WLANs with several APs are increasingly centrally managed to allow for a better use of their resources (e.g., channel allocation, assigning users to the APs, authentication). Besides proprietary solutions, standardized protocols like IETF's CAPWAP and IEEE 802.11v have been developped. These latter enable the exchange of information about the network topology and radio environment (within a WLAN) either from APs to a central controller, in the case of CAPWAP, or between users and APs, in 802.11v. However, the algorithms run by the controller and APs, and exploiting this knowledge, are yet to be designed. Their objectives are diverse and can range from estimating the actual throughput of APs given the network topology to reshuffling the channel allocation scheme. Note that the design of efficient algorithms must cope with the complexity brought by the CSMA/CA scheme used in the 802.11 MAC layer (e.g., random backoff, hidden node problem, starvation).

In this paper, we propose a Markovian, conceptually simple, and computationally efficient modeling approach to discover the behavior of each AP within a WLAN. More specifically, our approach returns an accurate estimation of the output rate (attainable throughput) of each AP given their respective load and the WLAN topology (conflict graph). The outcomes of our approach are manifold: gaining insight into an abnormal WLAN behavior, predicting WLAN performance under new settings, or reconfiguring an existing WLAN. We illustrate two possible applications where we rely on our model to make a better use of WLAN resources.

The rest of this paper is organized as follows. The next section briefly reviews the existing related works. Section 3 describes the system under study. The proposed modeling approach and the associate algorithm are presented in Section 4. Numerical results validating our approach and illustrating possible applications are discussed in Section 5. Finally, Section 6 concludes this paper.

II. State of the Art

In the last two decades, the performance analysis of the Distributed Coordination Function (DCF) used in IEEE 802.11 wireless networks has been an often revisited topic. Two of the pioneering papers in the field are the works of Bianchi [1] and Cali, Conti, and Gregori [2]. Both of them model the network at a very fine level of abstraction, taking into account the behavior of each individual packet transmission according to the CSMA/CA principles. In [2], the authors study the ratio of the average packet size and the average time it takes to transmit a packet, including all the protocol overhead. They apply this approach to study the utilization of the network's capacity for different numbers of contending stations and packet sizes.

Bianchi [1] introduces a two dimensional Markov chain to explicitly model the backoff process happening before each transmission of a packet in DCF in a fully-connected network (in which at most one node can be transmitting at a given time). A common property of these two models is that they were designed to handle only saturated networks. In a saturated network, all nodes have a packet waiting to be sent at all times, i.e., they have backlogged queues. Although saturation is a strong and often limiting assumption, these two works provide some of the ground research in the evaluation of wireless networks.

Bianchi's work has since been extended to non-saturated networks using a few different methods. Felemban and Ekici [3] have removed the condition of saturation by introducing the probability that a node has a packet waiting to be sent. To this aim, they create a second Markov chain, which is embedded into Bianchi's original chain [1]. This embedded chain describes the current state of the channel, which can be either idle, in collision, or in successful transmission. The solution to their model is found by successively iterating between the two chains. Upon convergence, the found solution delivers the steady state transmission probability for each node, which can then be used to evaluate the network's performance.

A different approach for relaxing the saturation constraint is given in [4] and [5], where a new state is added to Bianchi's Markov chain in order to represent a node that has an empty buffer. Similarly to Bianchi's original work, the scope of application for these two models is restricted to fully-connected networks. Note that Kosek-Szott's model [4] can also be used when users have heterogeneous traffic demands, creating a coexistence of low and high traffic densities inside the network.

In all these works, a fine-grained point of view of the network was taken, with a detailed description of each node. Another approach consists in evaluating the performance of a network from a higher level of abstraction. Two such models, that address non-saturated, multi-hop networks, are given in [6] and [7]. In a multi-hop network, a packet from node A travels across relay nodes before arriving at its destination node B (as opposed to single-hop networks, where A and B directly exchange packets). Both papers present two-level modeling approaches of unsaturated multi-hop wireless networks, in which the low-level model is a version of Bianchi's original Markov chain, while the high-level model aims at capturing the inter-node dependencies in the network. The solution to the overall model is found using a fixed-point iteration between the high and low levels. In [7], the high-level model consists of a set of *M/M/1/K* queues, where each queue represents a given node of the network. Although their modeling framework was designed to handle any number of nodes, examples shown in their paper involve multi-hop wireless paths with at most 4 nodes. In [6], the high-level model is a separate Markov chain describing the channel's behavior depending on the current states of neighboring nodes, with nodes being either idle, transmitting, or in backoff. Because of the three possible states for each node and the added complexity brought by multi-hop networks, the analytical model of [6] leads to a large state space as the number of nodes increases, making it intractable, for networks with more than 7 or 8 nodes, for which a decomposition into smaller networks is necessary.

In [8]–[10], Markov chains are used to model an entire network based on its topology. The states of the chain describe the set of nodes that are transmitting in the current network state. Nardeli and Knightly [8] rely on their proposed Markov chain to derive a model that takes into account the errors due to collisions and hidden terminals for a single-hop network. They come up with a closed-form multi-parameter expression of throughput, which is subsequently used for evaluating the performance of the considered network. Although the model accurately captures the behavior of CSMA/CA networks, it only deals with saturated networks and introduces some complexity due to the calculation of successful transmission probability. In [9], a similar Markov chain is used to evaluate the fairness and spatial reuse in multi-hop, saturated networks with different carrier sensing and reception ranges. More particularly, the authors study the spatial reuse in line-networks to show that CSMA/CA achieves maximal spatial reutilization as traffic intensity increases, at the cost of creating starvation in certain links. In [11], [12] CSMA/CA networks are modeled as continuous time Markov chains and the model is then used to study the fairness of the network. Jiang and Walrand [10] extend the usage of this model by proposing an adaptive solution that changes the nodes' backoff periods in the goal of maximizing the network's throughput and utilization.

A novel approach to modeling a non-saturated network is introduced in [13] and [14], where the authors have chosen to map the idle time of a node to a longer backoff period. This approach keeps the simplicity of a saturated network model by not explicitly representing idle states, and yet allows the study of unsaturated nodes. Laufer and Kleinrock [13] determine the throughput of a node in a CSMA/CA network using the ratio between the transmitting and the backoff periods of that node, its probability of successful transmission, and the channel capacity. The result is then used in the analysis of a network's capacity region, based on nodes' throughputs, under stability conditions. Bonald and Feuillet [15] also characterize both the capacity region and the stability of a wireless network. However, their work focuses on multi-channel networks in either ad-hoc or infrastructure mode, and they propose a refinement to CSMA to achieve a more efficient and fair access to the channel in the infrastructure mode.

In this paper, we study unsaturated, not fully-connected, single-hop IEEE 802.11 wireless networks. We present a conflict graph-based modeling approach to discover the attainable throughput of each node. Our model consists of a single Markov chain which aims at describing, at a high-level of abstraction, the current state of the entire wireless network. Unlike most existing works, our approach is conceptually and computationally simple, enabling it to easily cope with medium sized networks.

III. System Description

The considered system is a wireless network using the IEEE 802.11 DCF to access the radio channel. Nodes may represent

APs (Access Points), or user stations in infrastructure or ad-hoc modes. Either way, we study all nodes that belong to the same radio channel. A node detects, but does not necessarily successfully receive, data sent by nodes within its detection range. Throughout this paper, we refer to two nodes that belong to each other's detection ranges as *neighbors*. For the sake of simplicity, we assume that all nodes are sending their packets at the same speed (i.e., transmission rate).

The network under study consists of N nodes. We use a *conflict graph* to describe the interactions between the nodes. It is a simple graph of N vertices in which an edge exists between two vertices whenever, in the wireless network, the corresponding nodes belong to each other's detection ranges. Therefore, each pair of neighbor nodes in the network is connected by an edge in the conflict graph. Because we consider that all nodes have the same detection range, the edges in the conflict graph are undirected. Figure. 1 illustrates a simple example of a conflict graph for a four-node network.

Additionally, each node i, $(i = 1, \ldots, N)$, of the network is characterized by its *input rate*, denoted by x_i. The input rate refers to the node's demanded throughput, normalized by the link speed so that possible values for x_i range from 0 to 1. We assume that all x_i are mutually independent. Note that x_i can be regarded as the percentage of time node i demands access to the channel. Because a saturated node always has a packet waiting to be sent, its input rate is equal to 1.

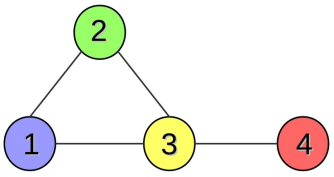

Fig. 1: Conflict graph for a four-node network.

As for the performance parameters of a wireless network, customary metrics include the end-to-end delay, the collision and loss rates, the resource utilization, and the attainable throughput. In this paper, we study the normalized attainable throughput (normalized by the link speed) of each node, which we refer to as its *output rate*. We let y_i denote the output rate of node i. In other words, y_i expresses the percentage of time in which node i is occupying the channel, i.e., is sending over the channel. It follows that, for $i = 1, \ldots, N$, $y_i \leq x_i$. The second performance parameter we are interested in is the *network utilization*, denoted by U and calculated as:

$$U = \frac{\sum_{i=1}^{N} y_i}{U_{max}}, \tag{1}$$

where U_{max} is the maximum network utilization. Otherwise stated, U_{max} is simply the maximum number of nodes that can be simultaneously sending, a quantity that clearly depends only on the conflict graph.

Table I summarizes the notation needed to describe the studied system, along with those pertaining to its corresponding model, presented in the next section.

N	total number of nodes
v_i	the set of neighbors of node i, $v_i \subseteq \{1, 2, ..., N\}$
w_i	the restricted set of neighbors of node i with synchronized, blocked, and preempted nodes removed
S	the set of possible sending states, $S \subseteq \{0, 1\}^N$
S_k	k-th sending state, $S_k \in S$
$S_k(i)$	state of node i in sending state S_k, $S_k(i) \in \{0, 1\}$, where 1 means node i is sending
A	the set of possible activity states, $A = \{\text{ON, OFF}\}^N$
A_k^t	an activity state associated to sending state S_k, $A_k^t \in A$
A_{kl}	the set of all activity states associated to sending state S_l and compatible with sending state S_k
$P_{A_k^t}$	probability of occurrence of activity state A_k^t
T	the set of possible transitions between sending states
T_{kl}	transition from sending state S_k to S_l, $T_{kl} \in T$
P_{kl}	probability of transition from sending state S_k to state S_l
x_i	input rate of i-th node, $x_i \in [0, 1]$
y_i	output rate of i-th node, $y_i \in [0, 1]$
U	network utilization, $U \in [0, 1]$
π	stationary probability distribution of the Markov chain that describes the network
π_{S_k}	steady state probability of sending state S_k

TABLE I: Notation.

IV. Modeling and Algorithm

A. Modeling approach

We now describe our high-level approach to modeling a wireless network as presented in the former section. To cope with the well-known complexity of CSMA/CA-based networks (e.g., random backoff, hidden node problem, starvation), we rely on a single Markov chain, but we deal with two intertwined state spaces, each describing differently and approximately the current state of the network. For the purpose of readability, we illustrate each step of our approach with an example, whose conflict graph is depicted in Fig. 1.

Our foremost state space, denoted by S and referred to as the *sending state* space, describes which nodes among the N are currently sending, and hence occupying the channel. We denote each sending state of S by S_k so that: $S = \{S_k\}_{k \geq 1}$. Therefore, each sending state S_k is a vector of size N in which the i-th value, $S_k(i)$, is set to 1 if node i is currently sending, and to 0 otherwise. Note that, in our work, a node's sending period includes the data packet transmission itself, as well as all protocol overhead (SIFS and DIFS intervals and ACK transmission). Our objective is to derive a Markov chain representing the possible transitions between these sending states, to compute the probabilities of its state transitions, i.e., its transition matrix, and finally, to derive the steady-state probabilities of S_k.

Because we deal with an unsaturated network, we need to introduce a secondary state space, referred to as the *activity state* space and denoted by A, that describes the activity of each node as being either ON or OFF. A node whose activity is ON has at least one packet waiting to be sent, while an OFF node has currently no backlogged packets. Analogously to S, each activity state, A_k^t, is a vector of size N whose i-th value denotes the activity of node i. Let $P_{A_l^u}$ denote the (steady-state) probability that the activity state A_l^u occurs. The mutual independence of the nodes' input rates x_i enables us to obtain $P_{A_l^u}$ as follows:

$$P_{A_l^u} = \prod_{A_l^u(i)=\text{ON}} x_i \prod_{A_l^u(j)=\text{OFF}} (1 - x_j). \tag{2}$$

Note that a sending state, S_k, can typically be associated to one or up to T different activity states, each denoted by A_k^t with $t = 1, \ldots, T$. Indeed, whereas a sending node necessarily implies that its activity is ON, the inverse does not hold. A node that is not sending and has no sending neighbors is (undoubtedly) OFF. On the other hand, if it does have a sending neighbor, then its activity can be either ON or OFF. This means that, when a node is not currently sending, it can be simply because it does not have a packet to be sent, or because it is waiting to access the channel. For example, assuming a given sending state $S_k = [\ 0\ 1\ 0\ 1\ ]$ for our network of four nodes, a total of four activity states ($T = 4$) may be associated:

$$\begin{aligned} A_k^1 &= [\ \text{OFF ON OFF ON}\ ] \\ A_k^2 &= [\ \text{ON ON OFF ON}\ ] \\ A_k^3 &= [\ \text{OFF ON ON ON}\ ] \\ A_k^4 &= [\ \text{ON ON ON ON}\ ] \end{aligned} \tag{3}$$

1) Determining Sending States: An intrinsic property of 802.11-based networks is that two neighbor nodes cannot (or virtually not) be transmitting simultaneously. In terms of modeling, this means that a sending state of the network is possible if and only if it does not include two transmitting vertices connected by an edge in the conflict graph.

In the case of our sample network represented in Fig. 1, out of a total of 16 sending states, only the following seven remain possible for the purpose of our modeling approach:

$$\begin{aligned} S_1 &= [\ 1\ 0\ 0\ 1\ ] & S_5 &= [\ 0\ 1\ 0\ 0\ ] \\ S_2 &= [\ 0\ 1\ 0\ 1\ ] & S_6 &= [\ 0\ 0\ 0\ 1\ ] \\ S_3 &= [\ 0\ 0\ 1\ 0\ ] & S_7 &= [\ 0\ 0\ 0\ 0\ ] \\ S_4 &= [\ 1\ 0\ 0\ 0\ ] \end{aligned} \tag{4}$$

2) Establishing Possible Transitions: Having defined the set of possible sending states, S_k, we need to decide when we allow a transition from a possible sending state, S_k, to another, S_l. This transition represents the fact that the network goes from state S_k to S_l upon a packet-sending completion. We apply a three-rule policy to determine if the transition from S_k to S_l is possible.

First, at most one element of S_k changes its value from 1 to 0 in S_l. This rule expresses that no more than one node stops sending at the same time (i.e., upon a change of state in the Markov chain). Considering our sample network, positive examples include transitions from S_1 to S_2 and from S_1 to S_4. Conversely, a negative example is the transition from S_1 to S_3.

Second, at most one element of S_k changes its value from 0 to 1 in S_l, unless in the exception case of a *synchronizing* node. A node i is said to be synchronizing if (i) it was sending in state S_k (i.e., $S_k(i) = 1$), and (ii) it has at least two neighbors that have an ON activity but are not neighbors themselves. The exception allows to account for the situation where a central node's end of sending triggers the beginnings of sending of several of its neighbor nodes. Said differently, in a case with two nodes, labeled m and n, and assuming that they have a common neighbor node i for which $S_k(i) = 1$ and $S_l(i) = 0$, we allow the transition from S_k to S_l even if we have $S_k(m) = S_k(n) = 0$ and $S_l(m) = S_l(n) = 1$. Apart from this exception, this second rule implies that no more than one node starts sending at the same time. In the case of our sample network, a positive example is the transition from S_4 to S_1 while a negative example is the transition from S_5 to S_1. The exception to the rule applies twice, namely when we allow the transitions from S_3 to S_1 and from S_3 to S_2.

Third, considering all potential activity states $\{A_k^t\}$ ($t = 1, \ldots, T$) associated to S_k and all potential activity states $\{A_l^u\}$ ($u = 1, \ldots, U$) associated to S_l, there must be a combination (A_k^t, A_l^u) in which at most one element changes its value from ON in A_k^t to OFF in A_l^u or from OFF to ON. A clear negative example is the transition from S_5 to S_6 (and vice versa) since this would require two nodes, i.e., the second and the fourth, to undergo a simultaneous change of activity.

Overall, the rationale behind these three rules resorts to the fact that we consider as negligible the probability that two nodes undergo simultaneous changes in their behavior (either in terms of sending or activity).

By applying this three-rule policy, we discover all the possible transitions between the sending states, i.e., the transition matrix, in which a value of 1 in the k-th row and l-th column indicates that the transition from S_k to S_l is possible, and has a value of 0 otherwise. We represent below an extract from the 7×7 transition matrix obtained when applying the three-rule policy on the sample four-node network:

$$\begin{array}{c} \\ S_1 \\ \vdots \\ S_3 \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccccccc} S_1 & S_2 & S_3 & S_4 & S_5 & S_6 & S_7 \end{array} \\ \left[\begin{array}{ccccccc} 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{array}\right] \end{array} \tag{5}$$

3) Computing Transition Probabilities: The next stage of our modeling approach is to find the probabilities of each transition. Let us consider the transition from S_k to S_l. First, we evaluate the number of activity states associated to S_l that are compatible with S_k. An activity state A_l^u is said to be *compatible* with S_k if there is at least one activity state associated to S_k that differs by at most one ON to OFF change (or vice versa) with A_l^u. We denote by A_{kl} the set of activity states associated to S_l that are compatible with S_k. For example, in our sample network, there are four activity states associated to S_1, namely:

$$\begin{aligned} A_1^1 &= [\ \text{ON OFF OFF ON}\ ] \\ A_1^2 &= [\ \text{ON ON OFF ON}\ ] \\ A_1^3 &= [\ \text{ON OFF ON ON}\ ] \\ A_1^4 &= [\ \text{ON ON ON ON}\ ] \end{aligned} \tag{6}$$

while those associated to S_2 are:

$$\begin{aligned} A_2^1 &= [\ \text{OFF ON OFF ON}\ ] \\ A_2^2 &= [\ \text{ON ON OFF ON}\ ] \\ A_2^3 &= [\ \text{OFF ON ON ON}\ ] \\ A_2^4 &= [\ \text{ON ON ON ON}\ ] \end{aligned} \tag{7}$$

We observe that all the activity states associated to S_2 are compatible with S_1, and thus belong to the set A_{12}, i.e., $A_{12} = \{A_2^1, A_2^2, A_2^3, A_2^4\}$.

Then, we simply derive the transition probability from S_k to S_l as:

$$P_{kl} = \sum_{A_l^u \in A_{kl}} (P_{A_l^u} \prod_{S_l(n)=1} \frac{1}{1 + \sum_{m \in v_n} \mathbb{1}_{\text{ON}(m)}}), \quad (8)$$

where v_n represents the set of neighbor nodes of node n. The sum $\sum_{m \in v_n} \mathbb{1}_{\text{ON}(m)}$ returns an integer value equal to the number of neighbors of node n that are ON in the activity state A_l^u.

Note that Eq. (8) captures that the transition probabilities depend both on the nodes' input rates and the conflict graph. For each activity state A_l^u that is compatible with the transition from the sending state S_k to S_l, we calculate the associated transition probability by first assuming that all concerned nodes are evenly likely to access the channel access, and then, we weigh the obtained value by the probability of occurrence of that activity state, $P_{A_l^u}$. The term $P_{A_l^u}$ in Eq. (8) ensures that the presence of a node with a high input rate will increase the probability that the whole network is in a sending state in which that node is sending. On the other hand, the other term within the sum represents the fact that a node that is well connected in the conflict graph experiences a lot of competition for channel access, which in turn diminishes its probability of gaining that access.

We now present two refinements to Eq. (8) that may occur, or not, depending on the network topology. Considering a given sending state S_k, some nodes may be viewed as blocked or as preempted by their neighbors. To cope with these nodes, we simply identify them and remove them from the considered neighbors of node n, i.e., $v(n)$, when using Eq. (8). A node is said to be *blocked* when it cannot start sending because it has two (or more) sending neighbors. For instance, node 3 is an example of a blocked node when our sample network is in sending state $S_1 = [\ 1\ 0\ 0\ 1\ ]$. Indeed, it is very unlikely for node 3 to start sending, as this would require nodes 1 and 4 to simultaneously stop sending. Analogously, a node is said to be *preempted* if it has one neighbor that is sending, and at least one other neighbor that is ON, which itself has no sending neighbors. This pertains to node 3 when our sample network is in sending state $S_4 = [\ 1\ 0\ 0\ 0\ ]$ with an activity state [ON OFF ON ON]. Under these circumstances, node 4 does not need to wait for the end of node 1's sending in order to start sending, which in turn means node 3 is likely to be preempted. In our modeling approach, we simply account for these nodes by phasing them out from the actual neighborhood of node n when considering the sending state S_k. In other words, in Eq. (8), we replace the set of neighbor nodes of n, i.e., $v(n)$, by the restricted set of neighbor nodes which excludes blocked and preempted nodes.

The other refinement refers to the case when the z-th node of S_k is a synchronizing node (whose definition was stated in Section IV.A.2) and continues to have an ON activity in the next activity state A_l^u associated to S_l. First, for each activity state associated to S_l, i.e., A_l^u, we estimate the probability that the synchronizing node continues to send in S_l using:

$$P_{z_l^u} = \frac{1}{1 + \sum_{m \in v_z} \mathbb{1}_{\text{ON}(m)}}, \quad (9)$$

where the sum $\sum \mathbb{1}_{\text{ON}(m)}$ returns an integer value equal to the number of neighbors of node z that are ON in activity state A_l^u. For example, considering the sending state S_3 of our sample network, node 3 is a synchronizing node. A possible activity state associated to S_3 is [ON OFF ON ON] in which node 3 has two neighbors that are also ON (i.e., nodes 1 and 4). Using Eq. (9), we obtain: $P_{z_l^u} = \frac{1}{3}$. The second step consists in refining the computation of the transition probability P_{kl} as follows:

$$P_{kl} = \sum_{A_l^u \in A_{kl}} (P_{A_l^u} \prod_{S_l(n)=1} \frac{1}{1 + \sum_{m \in w_n} \mathbb{1}_{\text{ON}(m)}} (1 - \mathbb{1}_{\text{z sync}}.P_{z_l^u})), \quad (10)$$

where $\mathbb{1}_{\text{z sync}}.P_{z_l^u}$ returns the probability $P_{z_l^u}$ if a synchronizing node z exists in node n's neighborhood, and 0 otherwise, and w_n denotes the restricted neighborhood of node n (without blocked, preempted, and synchronizing nodes).

Finally, we ensure that the matrix is row-stochastic by dividing each P_{kl} probability with the corresponding sum $\sum_{m=1}^{N} P_{km}$.

4) Deriving the Output Rates: Having evaluated all the transition probabilities between the sending states, we can easily obtain the steady-state probability of each sending state S_k. Let us denote by the vector $\pi = [\pi_{S_1}, \pi_{S_2}, ...]$ the corresponding steady-state probabilities. We then evaluate the output rate of node i by summing the steady-state probabilities π_{S_k} over the set of sending state S_k in which node i is sending. Hence, we have:

$$y_i = \sum_{k;\ S_k(i)=1} \pi_{S_k}. \quad (11)$$

In our sample network, we notice that node 1's is sending only in sending states S_1 and S_4. Thus, its output rate is given by: $y_1 = \pi_{S_1} + \pi_{S_4}$.

We conclude by briefly discussing the main approximations involved in our modeling approach. First, in our model, we do not allow two neighbor nodes to simultaneously access the channel. This implies that we neglect the potential collisions, and a fortiori their effect on the network performance. Second, our approach relies on a homogeneous Markov chain in which the transition probabilities are kept constant. In a real CSMA/CA network, owing to the backoff mechanisms, nodes do not have constant odds to access the channel. However, despite these simplifying approximations, our approach provides accurate results as discussed in the next section.

B. Algorithm

Algorithm 1

1: **Sending States:** Find all possible sending states:
2: $S = \{0, 1\}^N$
3: **for** $k = |S| ...1$ **do**
4: **if** S_k contains two sending neighbor nodes **then**
5: remove S_k from S
6: **end if**
7: **end for**

Activity States:
for $S_k \in S$ **do**
    Find all the activity states associated to S_k as described in Section IV-A
end for
Blocked nodes:
For any node i in a network state S_k where $S_k(i) = 0$:
if $\exists m, n \in \{v_i\}^2$ s.t. $S_k(m) = S_k(n) = 1$ **then**
    node i is a *blocked* node in state S_k
end if
Preempted nodes:
For any node i in a network state S_k and its activity state A_k^t where $S_k(i) = 0$:
if $\exists m, n \in \{v_i\}^2, m \notin v_n$ s.t. $S_k(m) = 1$ **and** $A_k^t(n) = ON$ **and** $\nexists r \in v_n$ s.t. $S_k(r) = 1$ **then**
    node i is a *preempted* node in state S_k for activity state A_k^t
end if
Synchronized nodes:
For any node i in a network state S_k where $S_k(i) = 1$ and an activity state A_k^t:
if $\exists m, n \in \{v_i\}^2, m \notin v_n$ s.t. $A_k^t(m) = A_k^t(n) = ON$ **then**
    node i is a *synchronizing* node for nodes m and n
end if
Transitions: Find the possible network state transitions
for $k = 1...|S|; l = 1...|S|$ **do**
    if $S_k \rightarrow S_l$ implies more than one $1 \rightarrow 0$ change in the network state **then**
        transition T_{kl} is not possible
    else if $S_k \rightarrow S_l$ implies more than one $0 \rightarrow 1$ change in the network state **except** for having up to n such changes in the n nodes synchronized by the same node **then**
        transition T_{kl} is not possible
    else if there is no state A_l^u compatible with S_k **then**
        transition T_{kl} is not possible
    else
        transition T_{kl} is possible
    end if
end for
Probabilities: Calculate all transition probabilities
for $T_{kl} \in T$ **do**
    Calculate activity state probabilities using (2)
    If synchronizing nodes exist, calculate $P_{z_l^u}$ using (9)
    Calculate the probability P_{kl} of T_{kl} with (10)
end for
Normalize:
    Normalize each transition probability P_{kl} by diving it with the sum of all transition probabilities of transitions leaving state S_k.
Solve the chain: Find the stationary distribution π of the Markov chain created by the states in S and the transitions in T
Calculate output rates: Use (11) to calculate the output rates of all N nodes.

V. Numerical Results

We begin this section with a study of our model's accuracy in predicting output rates. Then, we present two examples of possible applications. Throughout this section, all results were obtained using the IEEE 802.11g standard with a link speed of 54 Mbps. All nodes have equal detection ranges and transmit packets with a fixed size of 500 bytes.

A. Accuracy validation

We evaluate the accuracy of our model by comparing its values with those provided by the discrete-event simulator *ns2* [16]. Every simulation point presented in the results is the average of 20 independent simulation runs lasting 60 seconds each. The resulting confidence intervals are very narrow so we only use the mid-point in our validation. As for the input rates x_i, we represent their intensity in the simulator by alternating between active and inactive periods with exponentially distributed lengths whose mean values ratio is set to x_i. In active periods the node's queue is constantly backlogged, while in inactive period its queue is empty and the node is not attempting to access the channel. The throughput obtained in simulation is then normalized by the link speed, in order to obtain the dimensionless output rate, y_i, which is comparable with that of our model.

In our first example, we consider the four-node network of Fig. 1. We set the input rates of nodes 1, 3, and 4 to $x_1 = 0.5$, $x_3 = 1$, $x_4 = 0.5$, respectively. On the other hand, we let the input rate of node 2 vary from $x_2 = 0$ (always off) to $x_2 = 1$ (constant saturation), taking several values in that interval. We then evaluate the output rates of nodes 1 and 2, i.e., y_1 and y_2, as well as the network utilization, as delivered by the simulator and by our model. Figure. 2 shows the corresponding results.

As expected, as the input rate of node 2, i.e., x_2, grows from 0 to 1, so does its output rate, i.e., y_2, though to a lesser extent. Because $y_2 < x_2$ (except for $x_2 = 0$), we conclude that node 2 cannot meet all its demands, regardless of their actual intensity. As for node 1, whose input rate x_1 is kept constant, its output rate tends to decrease with increasing values of x_2. This behavior ensues from the competition between nodes 1 and 2. Note that, even for $x_2 = 0$, node 1 struggles to meet its demands as $y_1 = 0.4$ (while $x_1 = 0.5$). Finally, the network utilization grows steadily from around 0.6 to 0.75 as the input rate of node 2 increases. Overall, we observe that our proposed model is able to accurately capture these behaviors with a relative error typically less than 10%.

The second example involves a more complex network with a nine-node topology whose conflict graph is given in Fig. 3. Here, we consider several values for the input rate for node 6, x_6, ranging from 0 to 1 while we keep the input rates of other nodes constant, i.e., $x_1 = 0.7$, $x_2 = 0.8$, $x_3 = 0.6$, $x_4 = 0.5$, $x_5 = 0.8$, $x_7 = 0.7$, $x_8 = 0.6$, and $x_9 = 0.9$.

The corresponding results are shown in Fig. 4. In order to keep the figure legible, we choose to trace the output rates of nodes 5, 6, 7, and 9, bearing in mind that node 8's output rate evolves similarly to that of node 7, and that the first four nodes of the network are not highly influenced by the change in input rate of node 6. We observe that, as x_6 grows from 0 to 1, the behaviors of nodes can be significantly changed,

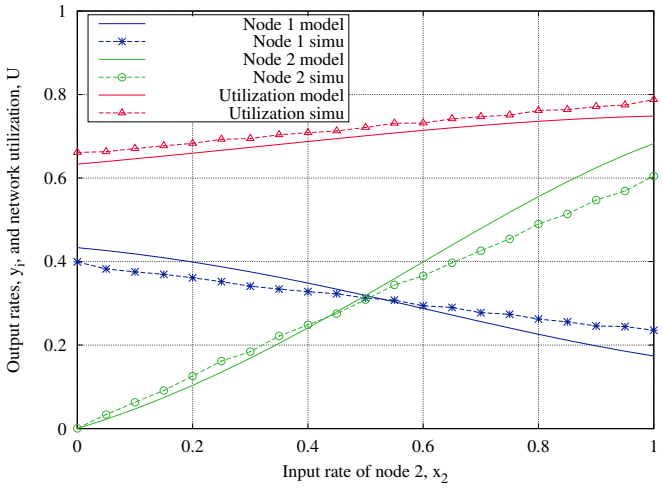

Fig. 2: Accuracy for the network of Fig. 1 as a function of node 2's input rate.

with some increasing their output rates (e.g., nodes 6 and 9), while others experience a decrease (e.g., nodes 5 and 7), that may result in a near node starvation as x_6 comes closer to 1. As shown by Fig. 4, our model was able to reproduce these non-trivial behaviors with a good degree of precision.

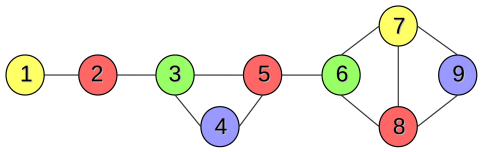

Fig. 3: Conflict graph for a nine-node network.

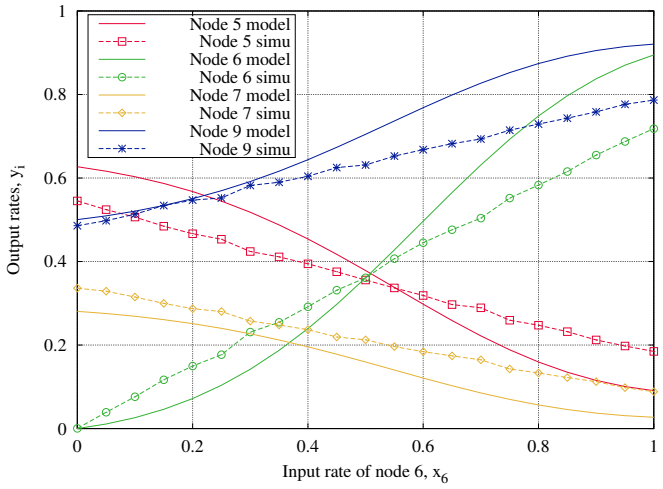

Fig. 4: Accuracy for the network of Fig. 3 as a function of node 6's input rate.

In the aim of providing a more comprehensive idea of the accuracy of our model, we present, in Table II, the overall error distribution obtained on over 250 samples. Each sample depicts the precision of the output rate of a node in the nine or four-node network. The median value indicates that, in 50% of the samples, the deviation between y_i as returned by the simulator and by our model is less than 0.047. Table II also shows that, in all our samples, the difference between the model's and the simulator's y_i is never larger than 0.20.

Median	**<0.05**	**0.05-0.1**	**0.1-0.2**	**>0.2**	**Utilization median**
0.047	52.7%	33.3%	14%	0%	0.053

TABLE II: Distribution of the absolute errors for the output rates, y_i.

Note that in order to investigate the robustness of our approach, we explored several other examples with different network topologies, packet sizes, and means and distributions for the active and inactive periods. Due to space constraints, the corresponding results are not presented in this paper. However, it was our experience that our model's accuracy was similar to that described by the two former examples.

B. Model applications

Having validated the accuracy of our model, we now describe how its application can help to improve the general behavior of a network by ensuring a better share of the channel resources. In the two scenarios that follow, we assume the nodes to be APs. Furthermore, we assume the presence of a network controller possessing the knowledge of the nodes' input rates and the network topology. We show how such a controller can be used to either restrain a node's input rate, or to completely turn off a node, in the goal of improving the network's performance.

In our first scenario, we suppose that the controller is able to regulate the input rate of a given node. Let us consider the nine-node network of Fig. 3 and that node 9 is selected. Note that node 9 plays a central role in the network topology as it is able to block or preempt nodes 7 and 8 with the help of node 6. Using our model, we discover the output rates of all nodes (including y_9) for values of x_9 ranging from 0.2 to 1, and we compute the associated value of Jain's index [17], a commonly used metric of fairness revealing how close the y_i values are to each other. Figure 5 reports the corresponding results. As expected, we observe that, as we relax the constraint on x_9, nodes 7 and 8 undergo a significant decrease in their output rates, unlike node 6 that experiences a growth of its output rate. More interestingly, the network's Jain's index exhibits a peak value for x_9 close to 0.35. This peak is about 30% higher than the value obtained with $x_9 = 1$. In other words, to ensure a fair share of the channel resources among the nodes, the best scheme would be to restrain the input rate of node 9 to no more than $x_9 = 0.35$.

Our second scenario addresses the case in which the network controller is entitled to completely turn off a node (e.g., by moving it to another radio channel). Our goal is to show how our model can help determine which node should be turned off in order to maximize a given performance metric (Jain's index or the network utilization U). To that effect, we consider the ten-node network whose conflict graph is given in Fig. 6 in which the input rates have been set to $x_1 = 0.3$, $x_2 = 0.6$, $x_3 = 0.8$, $x_4 = 0.5$, $x_5 = 0.7$, $x_6 = 0.3$, $x_7 = 0.5$,

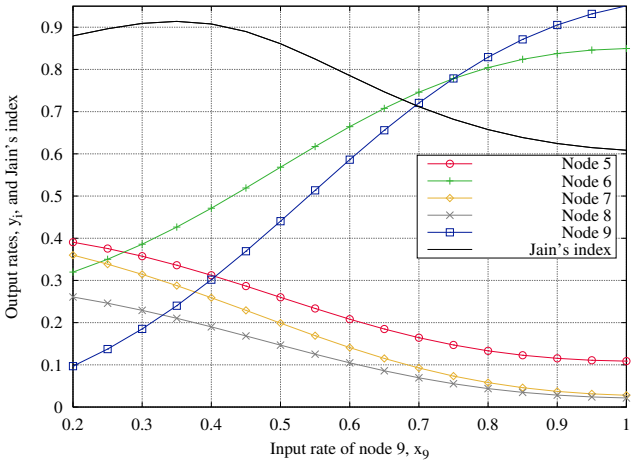

Fig. 5: Jain's fairness index and output rates for the network of Fig. 3 as a function of node 9's input rate.

$x_8 = 0.4$, $x_9 = 0.9$, and $x_{10} = 0.7$. One at a time, we turn off each node of the network, and then we calculate the nodes' output rates, together with the Jain's index and the network utilization. The corresponding results are shown in Fig. 7. Note that Ni denotes the case where the i-th node is turned off, while Original refers to the original ten-node network without any nodes off. It appears that turning node 3 off would be the best option as not only does it allow to achieve the largest network utilization, but also maximizes the Jain's fairness index.

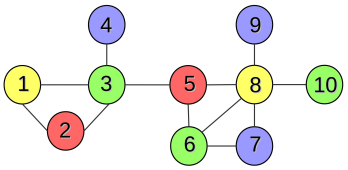

Fig. 6: Conflict graph for a ten-node network.

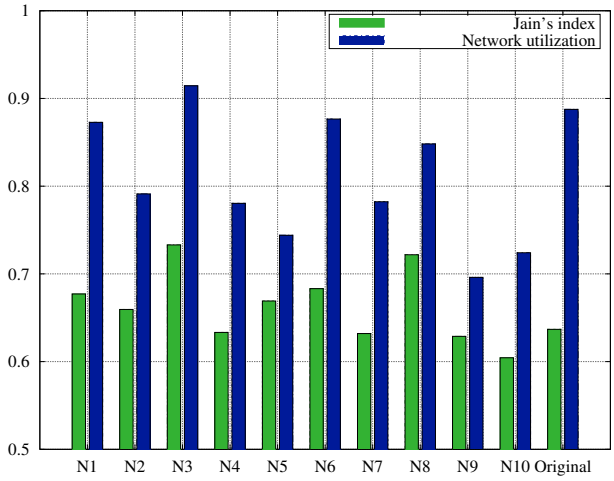

Fig. 7: Jain's fairness index and network utilization for the network in Fig. 6

VI. Conclusions

In this paper, we consider the performance evaluation of unsaturated, and not fully-connected networks based on IEEE 802.11 DCF. We present a conflict graph-based modeling approach to discover the attainable throughput of each node. Our model consists of a single Markov chain which aims at describing, at a high-level of abstraction, the current state of the entire wireless network. Owing to its low complexity, our approach is simple to implement, can cope with medium sized networks, and its execution speed is fast. We validate its accuracy against a discrete-event simulator. Results show that our approach is typically accurate, with associated relative errors generally less than 15%, and that it captures complex phenomena such as node starvation. We investigate two potential applications of our proposed approach in which, starting with a given network, we improve its performance in terms of overall throughput or fairness by throttling the throughput demand of a node, or by turning a node off altogether. Future works aim at validating our model against real-life measurements.

References

[1] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on selected areas in communications*, vol. 18, no. 3, pp. 535–547, 2000.

[2] F. Cali, M. Conti, and E. Gregori, "IEEE 802.11 wireless LAN: capacity analysis and protocol enhancement," in *INFOCOM'98. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1, pp. 142–149, IEEE, 1998.

[3] E. Felemban and E. Ekici, "Single hop IEEE 802.11 DCF analysis revisited: Accurate modeling of channel access delay and throughput for saturated and unsaturated traffic cases," *IEEE Transactions on Wireless Communications*, vol. 10, no. 10, pp. 3256–3266, 2011.

[4] K. Kosek-Szott, "A comprehensive analysis of IEEE 802.11 DCF heterogeneous traffic sources," *Ad Hoc Networks*, vol. 16, pp. 165–181, 2014.

[5] N. Gupta and C. Rai, "New analytical model for non-saturation throughput analysis of IEEE 802.11 DCF," in *International Conference on Advances in Communication, Network and Computing*, pp. 66–76, 2014.

[6] Z. Shi, C. Beard, and K. Mitchell, "Analytical models for understanding space, backoff, and flow correlation in CSMA wireless networks," *Wireless networks*, vol. 19, no. 3, pp. 393–409, 2013.

[7] T. Begin, B. Baynat, I. G. Lassous, and T. Abreu, "Performance analysis of multi-hop flows in IEEE 802.11 networks: A flexible and accurate modeling framework," *Performance Evaluation*, vol. 96, pp. 12–32, 2016.

[8] B. Nardelli and E. W. Knightly, "Closed-form throughput expressions for CSMA networks with collisions and hidden terminals," in *INFOCOM, 2012 Proceedings IEEE*, pp. 2309–2317, IEEE, 2012.

[9] M. Durvy, O. Dousse, and P. Thiran, "Self-organization properties of CSMA/CA systems and their consequences on fairness," *IEEE Transactions on Information Theory*, vol. 55, no. 3, pp. 931–943, 2009.

[10] L. Jiang and J. Walrand, "A distributed CSMA algorithm for throughput and utility maximization in wireless networks," *IEEE/ACM Transactions on Networking (ToN)*, vol. 18, no. 3, pp. 960–972, 2010.

[11] R. Boorstyn, A. Kershenbaum, B. Maglaris, and V. Sahin, "Throughput analysis in multihop csma packet radio networks," *IEEE Transactions on Communications*, vol. 35, no. 3, pp. 267–274, 1987.

[12] X. Wang and K. Kar, "Throughput modelling and fairness issues in CSMA/CA based ad-hoc networks," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 1, pp. 23–34, IEEE, 2005.

[13] R. Laufer and L. Kleinrock, "The capacity of wireless CSMA/CA networks," *IEEE transactions on Networking*, 2015.

[14] C. Kai and S. Zhang, "Throughput analysis of CSMA wireless networks with finite offered-load," in *IEEE International Conference on Communications (ICC)*, pp. 6101–6106, IEEE, 2013.

[15] T. Bonald and M. Feuillet, "Performance of CSMA in multi-channel wireless networks," *Queueing Systems*, vol. 72, no. 1-2, pp. 139–160, 2012.

[16] *The Network Simulator ns-2*. http://www.isi.edu/nsnam/ns/.

[17] R. Jain, D. M. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer system," *Digital Equipment Corporation*, 1984.