A New Modelling Approach to Represent the DCF Mechanism of the CSMA/CA Protocol

Marco Scarpa(✉) and Salvatore Serrano

Department of Engineering, University of Messina,
Contrada di Dio, 98166 S. Agata, Messina, Italy
{mscarpa,sserrano}@unime.it

Abstract. In this paper, a Markovian agent model is used to represent the behavior of wireless nodes based on CSMA/CA access method. This kind of network was usually modeled by means of bidimensional Markov Chains and more recently using semi-Markov process based models. Both these approaches are based on the assumptions of both full load network and independence of collision probability with respect to retransmission count of each packet. Our model inherently releases the latter hypothesis since it is not necessary to establish a constant collision probability at steady state.

Here, we investigate the correctness of our approach analyzing the throughput of a network based on two IEEE 802.11g nodes when the amount of traffic sent by each one varies. Results have been compared with Omnet++ simulations and show the validity of the proposed model.

1 Introduction

In this paper, we propose a new model for the study of multi-hop Carrier Sense Multiple Access with collision avoidance (CSMA/CA)-based networks. The CSMA/CA protocol is used in a great number of wireless networks to access the medium at MAC layer level. For example the IEEE 802.11 employs a CSMA/CA mechanism with binary exponential backoff (BEB) rules, called Distributed Coordination Function (DCF). DCF defines a basic access method, and an optional four-way handshaking technique, known as request-to-send/clear-to-send (RTS/CTS) method [1]. Even if in this paper we address the basic access mechanism, the approach here presented is promising to be extended in order to take into account the RTS/CTS mechanism. We provide a powerful model based on Markovian agents that accounts for all the exponential backoff protocol details, and allows us to compute throughput performance of DCF for the basic access mechanisms. A more realistic modeling of the mechanism is possible by means of Markovian agents because the relative load condition and retransmission status of each wireless node in the network is taken into account. Practically, using our approach, it is possible to release the key approximation assumed in [2], and subsequent works [3–5] related to the assumption of constant and independent collision probability of a transmitted packet regardless of the number

© Springer International Publishing AG 2017
N. Thomas and M. Forshaw (Eds.): ASMTA 2017, LNCS 10378, pp. 181–195, 2017.
DOI: 10.1007/978-3-319-61428-1_13

of already done retransmissions. In other words, our model will be capable to intrinsically analyze saturated and unsaturated traffic cases [6,7].

The rest of the paper is organized as follows: Sect. 2 presents a summary of the most important papers dealing with modeling of the DCF mechanism; Sect. 3 summarizes the basic access method of DCF; Sect. 4 briefly summarizes the MA theory and introduces the MA model of a wireless node implementing the DCF mechanism; Sect. 5 presents the reference scenario, and the simulation environment we used to validate the MA model; moreover numerical results from both simulation and analytical solution of a MA based scenario are shown. Finally Sect. 6 reports conclusions.

2 Related Work

A great number of different models are introduced for the CSMA-based protocols, most of them, after Bianchi's seminal paper [2], are based on two-dimensional Markov models.

The problem is solved by Bianchi in two distinct steps. In the first step, by means of a Markov model, the behavior of a single wireless node, also called *station*[1], is studied. This model allows to obtain the probability that a station transmits a packet in a generic time slot. In the second step, the throughput is expressed as function of the probability evaluated in the previous step by analyzing the events that can occur within a generic time slot. Two independent processes are considered to model the BEB mechanism: $b(t)$ (the stochastic process representing the backoff time counter for a given station) and $s(t)$ (the stochastic process representing the backoff stage of the station at time t). A restrictive hypothesis is to consider the collision probability p constant regardless of the number of retransmissions that each packet suffers. Assuming the independence of this probability with respect to the number of retransmissions, it is possible to assume a constant value for the same probability p and to model the entire process by means of a bidimensional discrete-time Markov chain.

Recently, a novel Semi-Markov Process (SMP) based model for the single-hop IEEE 802.11 has been proposed to mitigate the complexity of the Bianchi's based model [3]. An SMP is a generalization of Markov chain. It includes a state holding-time that permits to increase the specification of the process. The holding-time of a state i is the amount of time spent before leaving state i. Differently by a traditional Markov process, the future state of a SMP depends not only on the current state but also on its state holding-time. The authors of [3] design the model in three different steps: (1) They build a $(m+1)$-state Markov Chain. The state i in this chain represents the i^{th} backoff stage of the BEB mechanism. An unsuccessful transmission is modeled by a transition from a lower state i to a higher state $(i+1)$. A successful transmission is modeled by a transition from any state i to state 0. Loopback transitions are possible only for states 0 and m. (2) The Markov chain defined in step (1) is transformed into an embedded Markov chain. An embedded Markov chain is characterized

[1] From this point on, we use the terms *station* and *wireless node* interchangeably.

by transition probability $P_{ii} = 0$, $\forall i$. The state holding-times of a discrete-time Markov chain are equal to a unit time and are independent of the next state transition. It is possible to consider the sample paths of a SMPs as timed sequences of state transitions. If one watches the process at the times of state transitions, the sample paths of the SMP are identical to those of a Markov chain. Such a process is known as embedded Markov chain. (3) The embedded Markov chain is transformed to an SMP in which the state holding-time for state i will be a random value uniformly selected within the range $(0, 2^i \cdot CWmin)$, for $0 \leq i \leq m$. Once the model is obtained, the authors use the stationary probability distribution of the SMP and the state holding-times to compute the packet transmission probability τ and the saturation throughput in the network. Indeed, the state holding-time for state i in the embedded Markov chain models the backoff interval of backoff stage i and the stationary probability Π_i^s of a SMP represents the fraction of time spent by a wireless node in backoff stage i. The proposed SMP model achieves accurate results with less complexity and computational time with respect to Bianchi's model. An advanced SMP model was proposed in [4]. It calculates the network parameters of single-hop WLANs more accurately. Some other studies also deal with the modeling of CSMA-based protocols in multi-hop scenarios [5,8]. In [9], the authors apply SMP modeling to linear multi-hop networks. The key assumption in all these models is that the probability p that a transmitted packet suffers a collision is the probability that at least one of the $N-1$ remaining stations transmit in the same time slot. Moreover, if each station transmits a packet with probability τ and it is also assumed the independence of collisions and retransmissions, it is possible to prove that, at steady state, $p = 1 - (1-\tau)^{N-1}$. In [6], an extension of the model proposed by Bianchi is presented with the aim to evaluate network load in saturation condition taking into account the channel state during the backoff countdown process. Moreover, the authors extend the model to unsaturated traffic cases through an iterative approach which allows to obtain accurate performance metric estimations for a wide range of parameters. Instead in [7], the authors derive analytical expressions for unsaturated network throughput of the IEEE 802.11 DCF using three different schemes: the physical-layer network coding, the traditional nonphysical-layer network-coding, and without network-coding. As mentioned above, in this work we introduce a model that releases all these assumptions and consequently allows to evaluate the behavior of a saturated and an unsaturated network, also considering different traffic loads for each node of the network.

3 The Distributed Coordination Function (Basic Access Method)

In this section, for the sake of completeness, we briefly explain the operation of the basic access method using the DCF mechanism in a IEEE 802.11g WiFi network. In a IEEE802.11g network, when the basic access mechanism is used, a station with packets to transmit monitors the channel activities until it observes the channel is busy. If an idle period equals to a distributed inter-frame space (DIFS) is detected, the station starts the random access mechanism implemented by the backoff period. This latter period is slotted so it is possible to express its length in terms of an integer number of elementary backoff slots. After sensing an idle DIFS, the station initializes a backoff counter. This counter will be decremented by one whenever a time slot expires while the channel is sensed idle. The counter will be stopped when the channel becomes busy due to a transmission of another station and reactivated when the channel will be sensed idle again for more than a DIFS. The station will begin the transmission of the packet when the backoff counter will reach zero. In this manner each station will wait for a random backoff period for an additional deferral time before transmission. If at least two stations decide to start transmission in the same time slot, a collision occurs. When the backoff mechanism starts, the backoff counter is uniformly chosen in the range $[0, CW]$, where CW is the current backoff contention window size. At the first attempt of each packet transmission, CW is set equal to the minimum contention window size CW_{min}. After each unsuccessful transmission (i.e. when a collision occurs), CW is doubled so the next backoff period could be chosen longer. The window will be doubled until the maximum contention window size CW_{max} will be reached. If the window size reaches CW_{max} and other collisions occur, CW shall remain at the value CW_{max}. The backoff contention window size will be reset to CW_{min} value either if a successful attempt to transmit occurs or if the retransmission counter reaches a predefined retry limit (the maximum retransmission number). When the retry limit is reached, the present packet is dropped. If the a packet transmission attempt is successful and the destination station successfully receives the packet, this one responds with an acknowledgment (ACK). The ACK will be followed by a short inter-frame space (SIFS) time during which the channel will be idle. If the transmitting station does not receive the ACK within a specified ACK_Timeout, or it detects the transmission of a different packet on the channel, it decides for collision and reschedules the packet transmission. Furthermore, after the reception of an error packet, a station shall be wait an extended inter-frame space (EIFS) before starting the transmission of a new packet.

In the literature, there have been considerable researches aiming at analyzing the performance of the DCF mechanism.

4 Markovian Agent to Model a DCF Node

In recent years, a new versatile analytical technique [10,11] has emerged whose main idea is to model a distributed system by means of interacting agents. This technique defines each agent through its local properties, but also introduces a mechanism in order to modify its own behavior according to the influence of the interactions with other agents. In this way, the analysis of each agent alone incorporates the effect of the inter-dependencies.

In this section, we briefly recall the basics on Markovian Agents Models and then we describe our proposal for representing the behavior of two interacting wireless stations.

4.1 Markovian Agents

Markovian Agent Models (MAMs) represent systems as a collection of agents scattered over a geographical space, and described by a continuous-time Markov chain (CTMC) where two types of transitions may occur: *local transitions* and *induced transitions*. The former models the internal features of the MA, whereas the latter accounts for interaction with other MAs. When a local transition occurs, an MA can send a message to other MAs. The propagation of messages is regulated by the *perception function* $u(\cdot)$. Depending on the agent position in the space, on the message routing policy, and on the transmittance properties of the medium, this function allows the receiving MA to be aware of the state from which the message was issued, and to use this information to choose an appropriate action. MAs can be scattered over a geographical area $\mathcal{V}$. Agents can be grouped in classes and can share different types of messages.

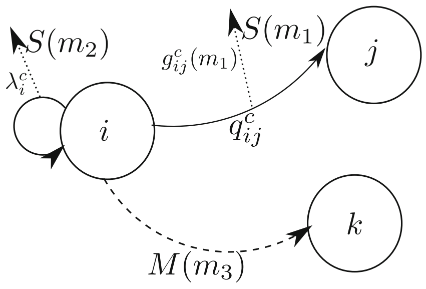

Fig. 1. Graphical representation of MAs

We represent a Markovian agent by exploiting the graphical notation like in Fig. 1. Given an MA of class c, a local transition from state i to state j is drawn as usual with a solid arc and, eventually, the associated rate q_{ij}^c. When a transition happens a message m could be sent with probability $g_{ij}^c(m)$; this event is graphically drawn as a dotted line starting from the transition whose firing sends the message and it is labeled with $S(m)$ to make evident which message is sent. An MA is also able to send a message during the sojourn in a particular state: self-loops are used to this aim; in fact, a self-loop in state i could be used to send a message m at a given rate λ_i^c similarly to what happens during a state transition. It is worth to note that self-loops do not influence local behavior of MAs, like in the usual theory of CTMCs, due to the memoryless property of the exponential distribution; instead they have a role in the evolution of a remote MA receiving the sent message. An example of MA is depicted in Fig. 1, where message m_1 is sent at the occurrence of transition from i to j, and a self-loop is associated with state i emitting message m_2 at rate λ_i^c. Induced transition due to reception of a message is graphically represented with a dashed arc between involved states; in this case the arc is labeled with $M(m)$. As an example, in Fig. 1, a transition from state i to state k, is due to the reception of message m_3.

Formally a *Multiple Agent Class, Multiple Message Type* Markovian Agents Model (MAM) is defined by the tuple:

$$MAM = \{\mathcal{C}, \mathcal{M}, \mathcal{V}, \mathcal{U}, \mathcal{R}\}, \tag{1}$$

where $\mathcal{C} = \{1 \dots C\}$ is the set of agent classes, and $\mathcal{M} = \{1 \dots M\}$ is the set of message types. $\mathcal{V}$ is the finite space over which Markovian Agents are spread, and $\mathcal{U} = \{u_1(\cdot) \dots u_M(\cdot)\}$ is a set of M perception functions (one for each message type). The density of the agents is regulated by functions $\mathcal{R} = \{\xi^1(\cdot) \dots \xi^C(\cdot)\}$, where each component $\xi^c(\mathbf{v})$, with $c \in \mathcal{C}$, counts the number of class c agents deployed in position $\mathbf{v} \in \mathcal{V}$. Since in this work the space is considered discrete, each position could be identified by a *cell* numbered with an integer with respect to some reference system.

Each agent MA^c of class c is defined by the tuple:

$$A^c = \{\mathbf{Q}^c, \mathbf{\Lambda}^c, \mathbf{G}^c(m), \mathbf{A}^c(m), \boldsymbol{\pi}_0^c\}. \tag{2}$$

Here, $\mathbf{Q^c} = [q_{ij}^c]$ is the $n_c \times n_c$ infinitesimal generator matrix of the CTMC that describes the local behavior of a class c agent, and its element q_{ij}^c represents the transition rate from state i to state j (and $q_{ii}^c = -\sum_{j \neq i} q_{ij}^c$). $\mathbf{\Lambda}^c = [\lambda_i^c]$, is a vector of size n_c whose components represent the rates at which the Markov chain reenters the same state: this can be used to send messages with an assigned rate without leaving a state. $\mathbf{G}^c(m) = [g_{ij}^c(m)]$ and $\mathbf{A}^c(m) = [a_{ij}^c(m)]$ are $n_c \times n_c$ matrices that represent respectively the probability that an agent of class c generates a message of type m during a jump from state i to state j, and the probability that an agent of class c accepts a message of type m in state i and immediately jumps to state j. $\boldsymbol{\pi}_0^c$, is a probability vector of size n_c which represents the initial state distribution.

The perception function of a MAM is formally defined as $u_m : \mathcal{V} \times \mathcal{C} \times \mathbb{N} \times \mathcal{V} \times \mathcal{C} \times \mathbb{N} \rightarrow \mathbb{R}^+$. The values of $u_m(\mathbf{v}, c, i, \mathbf{v}', c', i')$ represent the probability that an agent of class c, in position $\mathbf{v}$, and in state i, perceives a message m generated by an agent of class c' in position $\mathbf{v}'$ in state i'. Thanks to perception functions, a different instances of agents deployed over the space can interact sending messages one each others. Interactions are technically implemented through the matrix $\mathbf{\Gamma}^c(t, \mathbf{v}, m)$, a diagonal matrix collecting the total rate of received messages m by an agent of class c in position $\mathbf{v}$ (element γ_{ii} stores the value for state i). Matrix $\mathbf{\Gamma}^c(t, \mathbf{v}, m)$ is used to compute the infinitesimal generator matrix of class c agent at position $\mathbf{v}$ at time t: $\mathbf{K}^c(t, \mathbf{v}) = \mathbf{Q}^c + \sum_m \mathbf{\Gamma}^c(t, \mathbf{v}, m)\, [\mathbf{A}^c(m) - \mathbf{I}]$. The overall Markovian agent model thus evolves according the set of coupled differential equations

$$\frac{d\rho^c(t, \mathbf{v})}{dt} = \rho^c(t, \mathbf{v}) \mathbf{K}^c(t, \mathbf{v}) \tag{3}$$

under the initial condition $\rho^c(0, \mathbf{v}) = \xi^c(\mathbf{v}) \boldsymbol{\pi}_0^c$, $\forall \mathbf{v} \in \mathcal{V}$, $\forall c \in \mathcal{C}$.

As deeply described in [10,11], the main advantage of Markovian agent is that state space complexity is maintained low because dependencies between two agents are modeled through messages instead defining the cross product of their state spaces. Solution method to solve Eq. (3) uses discretization techniques for both time and space and fixed point based algorithms.

4.2 MA Classes in the Model

In the context of DCF, we use two classes of MAs to represent each wireless node: the *Buffer* and the *Backoff* classes. The two classes are depicted in Figs. 2 and 3 respectively.

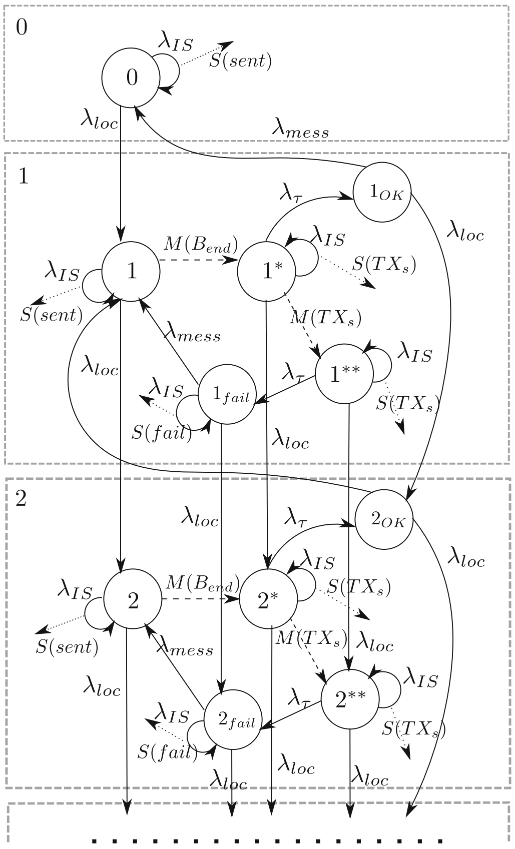

Fig. 2. *Buffer* class

The *Buffer* class models the behavior of wireless node with respect to the transmission of stored frames. It is constituted by $N + 1$ blocks, being N the buffer dimension of the wireless node. A generic block i in the class represents the wireless node buffer containing i frames. When some frames are buffered ($i > 0$), the wireless node tries to transmit one of them at the expiration of the backoff counter. To model this mechanism, we used five states. State i identifies the wireless node with i frames in the buffer waiting for the counter expiration; at the expiration, signaled by the message B_{end} sent by the Backoff class agent, the state changes into i^*. Since each MA of Buffer class sends a TX_s message

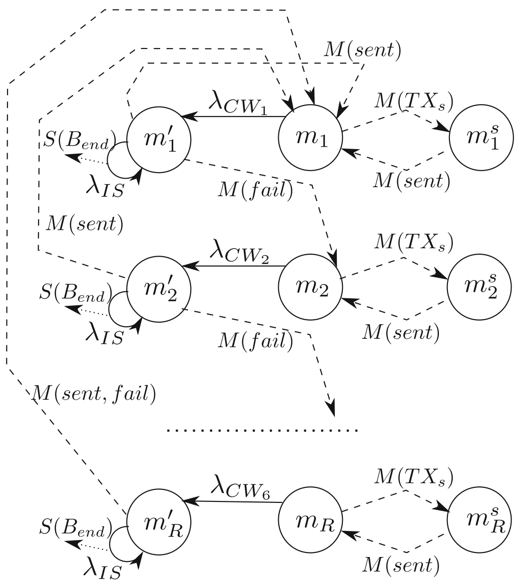

Fig. 3. *Backoff* class

when it is in a state attempting for frame transmission, all the others in the same condition can detect the collision transiting to state i^{**} induced by a perceived TX_s message. The states i^* and i^{**} allow us to discriminate whether the message transmission after the time slot expiration will result in a failure or not. When the transition happens from the former state (to state i_{OK}), the transmission will succeed otherwise (to state i_{fail}) it will fail due to a collision. The rate $\lambda_\tau = \frac{1}{\tau}$ reflects the duration of the time slot τ. In order to correctly signal to other MAs its own state, a Buffer class MA sends TX_s messages through self-loops in states i^* and i^{**}. The state $i-1$ is reached from i_{OK}, decreasing the number of buffered frames; instead, MA comes back to state i when a collision occurs. The sojourn time in i_{OK} and i_{fail} is the time t_{mess} to send a frame, thus we set the corresponding state transitions to $\lambda_{mess} = \frac{1}{t_{mess}}$. Messages $fail$ and *sent* are used to notify to the Backoff class MA the outcome of transmission attempt. Of course, irrespective of the state, a new packet to be transmitted could arrive from higher layer; to model this event, we connected each state in block i, with $0 < i \leq N-1$, with the equivalent in block $i+1$. We denote the frame arrival rate with λ_{loc}.

Block 0 has a different structure since it represents the empty buffer thus no transmission attempts are done and one state only (the state 0) completely represent the system. The message sent in 0 is used to trigger the start of Backoff class MA in the case of countdown expiration.

Based on this description, matrices storing transition rates and self-loop rates can be easily built. The *Buffer* class initial probability vector set to 1.0 the probability to be in state 0. Generating matrices and acceptance matrices are set in such a way the described messages are sent and received with probability 1.0. We do not write their complete structure for lack of space.

The Backoff class (Fig. 3) has the task of regulating the starting of transmissions of *Buffer* class agents by sending the B_{end} message. This is implemented with three states denoted in the following as m_i, m_i' and m_i^s. State m_i represents the system waiting for the backoff counter expiration in the contention window CW_i; when the contention timeout expires the MA transits to m_i'. The transition rate from m_i to m_i' is computed taking into consideration the contention windows at i-th attempt as follows:

$$\lambda_{CW_i} = \begin{cases} \frac{1}{2^{i-1}CW_{min}} & 2^{i-1}CW_{min} < CW_{max} \\ \frac{1}{CW_{max}} & otherwise \end{cases} \tag{4}$$

A B_{end} message is emitted in the states m_i' signaling to the perceiving MAs that the contention timeout expired.

All the other transitions in this class are induced by other MAs. In particular, when a *Backoff* class MA is in m_i' it can transit in either m_{i+1} or m_1 if the corresponding *Buffer* MA failed in transmitting the frame or not, respectively. These events are considered by perceiving the messages $fail$ and *sent* emitted by the MA modeling the frame transmission. In presence of more *Buffer* class MAs, only the messages generated by the MA in the same location of the Backoff MA are perceived. Instead transitions back and forth to m_i^s are due to messages emitted by Backoff class MAs in different locations of the receiving Buffer class MA. In fact a TX_s message has to be perceived if emitted by some transmitting stations, different by the actually considered, because it acquired the channel; thus the countdown is stopped (transition to m_i^s) and it will be resumed (transition back to m_i) at the reception of *sent* message.

A slightly different behavior is implemented in the state m_R', where a reception of $fail$ message reset the backoff algorithm to m_1 because the maximum number of retries has been reached.

As in the previous case, based on this description, matrices storing transition rates and self-loop rates can be easily built. The *Backoff* class initial probability vector set to 1.0 the probability to be in state m_1. Generating matrices and acceptance matrices are set in such a way the described messages are sent and received with probability 1.0. We do not write their complete structure for lack of space.

4.3 The Implemented *MAM*

In order to define the complete model we have to define all the quantities in (1).

In the case considered in this paper, the space $\mathcal{V}$ is a very little rectangular area composed of two cells. This choice is due to the fact that, in this work, we focus on correctly modeling the interaction mechanism and thus only two transmitting stations are considered. Due to this reason, the space $\mathcal{V}$ is simply defined by two positions and we have $\mathcal{V} = \{0, 1\}$.

The set $\mathcal{C}$ is defined by the two class introduced in Sect. 4.2: for the sake of simplicity, we will denote with "u" and "a" the *Buffer* and *Backoff* class respectively. Thus $\mathcal{C} = \{u, a\}$, whereas $\mathcal{M} = \{TX_s, B_{end}, fail, sent\}$.

Since each wireless node is modeled by a couple of MAs (one *Buffer* MA and one *Backoff* MA) and we considered two wireless nodes in the space, four interacting MAs are in the model. From these considerations, the set $\mathcal{R}$ easily derives being $\xi^u(0) = \xi^a(0) = \xi^u(1) = \xi^a(1) = 1$.

Perception functions $u_m(\cdot)$, with $m \in \mathcal{M}$, are specified in Table 1 where only the non null values are written. In this paper, we assumed that when a message is sent to a given target destination it is perceived, thus all the values of $u_m(\cdot)$ in the table are equal to 1.0. Perception functions reflect the interaction among MA classes as described in Sect. 4.2.

Table 1. MAM perception functions

	Perceiving MA			Receiving MA		
Message	v	Class	State	v'	Class	State
B_{end}	0	u	i, $1 \le i \le N$	0	a	m'_j, $1 \le j \le R$
B_{end}	1	u	i, $1 \le i \le N$	1	a	m'_j, $1 \le j \le R$
TX_s	0	u	i^*, $1 \le i \le N$	1	u	i^*, $1 \le i \le N$
TX_s	0	u	i^*, $1 \le i \le N$	1	u	i^{**}, $1 \le i \le N$
TX_s	0	a	m_j, $1 \le j \le R$	1	u	i^*, $1 \le i \le N$
TX_s	0	a	m_j, $1 \le j \le R$	1	u	i^{**}, $1 \le i \le N$
TX_s	1	u	i^*, $1 \le i \le N$	0	u	i^*, $1 \le i \le N$
TX_s	1	u	i^*, $1 \le i \le N$	0	u	i^{**}, $1 \le i \le N$
TX_s	1	a	m_j, $1 \le j \le R$	0	u	i^*, $1 \le i \le N$
TX_s	1	a	m_j, $1 \le j \le R$	0	u	i^{**}, $1 \le i \le N$
$fail$	0	a	m'_j, $1 \le j \le R$	0	u	i_{fail}, $1 \le i \le N$
$fail$	1	a	m'_j, $1 \le j \le R$	1	u	i_{fail}, $1 \le i \le N$
sent	0	a	m'_j, $1 \le j \le R$	0	u	i, $0 \le i \le N$
sent	0	a	m^s_j, $1 \le j \le R$	1	u	i, $0 \le i \le N$
sent	1	a	m'_j, $1 \le j \le R$	1	u	i, $0 \le i \le N$
sent	1	a	m^s_j, $1 \le j \le R$	0	u	i, $0 \le i \le N$

We note that the use of MAs makes simple the extension to a more complex scenario with N nodes: it is enough to deploy a *Buffer* MA and a *Backoff* MA for each wireless node and appropriately define the perception functions for each pair of deployed MAs. Since the purpose of this paper is to show the usefulness and the advantages of Mobile Agent modeling paradigm in representing interacting wireless stations, we consider only a two wireless nodes scenario leaving the study of more complex scenarios as future research.

5 Model Validation and Results

In this preliminary work, we referred to a simple IEEE 802.11g wireless network to evaluate the accuracy of our DSF model; we take into account the network throughput by varying the network load from a lightweight to a saturated one. The IEEE 802.11g operates in the 2.4 GHz ISM band and provides a maximum raw data throughput of 54 Mbps, although this translates to a real maximum throughput of 24 Mbps. In this simple scenario, we set two wireless nodes transmitting an UDP flow one each other. We did not consider any kind of noise or signal interference neither signal attenuation. The flows are characterized by an exponentially distributed UDP payload size. We evaluated model results using both an average value of 100 bytes and an average value of 1000 bytes for the UDP payload size. Moreover, we considered an exponentially distributed generation of information units at the application level.

According to the bidirectional traffic flow in each node, it is possible to evaluate the mean load of the network R, expressed in bps, by using the following relation

$$R = \frac{2 \cdot L}{g} \tag{5}$$

where L is the average frame length expressed in bits ($L = C \cdot t_{mess}$, with $C = 54$ Mbps the maximum raw data rate) and g is the average value of the intrapacket gap (IPG) ($g = \frac{1}{\lambda_{loc}}$). To validate our model, we also simulated the two interacting wireless nodes. We used "Omnet++" as a network simulator. For simulation purpose, we put a 802.11 g access point and a 802.11 g wireless host in a predefined squared area and we imposed a "stationary mobility type". Each device has been equipped with a NIC having a MAC and PHY using the parameters showed in Table 2. Accordingly, the simulator never uses the Request To Send/Clear To Send (RTS/CTS) mechanism because we imposed the maximum length of each packet to 1500 bytes, i.e. the Ethernet Maximum Transfer Unit (MTU). Moreover, considering the PHY parameters and the distance of the nodes, each transmitted packet will be received without errors due to the strictly high SNR. In this conditions only collisions can disrupt the reception of a packet. Finally, the UDP IPG (g) has been assumed to be an exponentially distributed random variable with mean value set accordingly to the desired load of the network. We performed 50 simulation runs for each evaluated point

Table 2. MAC & PHY parameters

Control bit rate	54 Mbps	Basic bit rate	54 Mbps
Max queue size	14 packets	RTS threshold	3000 bytes
Contention window (min size)	31 time slots	Retry limit	7 times
Contention window (max size)	255 time slots	Tx Power	20 mW
Receiver sensitivity	–85 dBm	Noise power	–192 dBm
Distance	10 m		

(a specified network load) each lasting 12 s. Confidence intervals obtained is very narrow, thus we did not show them in the related graphs.

Derivation of network throughput from the MAM model is obtained considering the probability a transmission successfully complete. This is easily evaluated considering the states i_{OK} of the *Buffer* class MAs as follows:

$$Th_f = \sum_{i=1}^{N} \left(P[M^u(0) = i_{OK}] + P[M^u(1) = i_{OK}] \right) \lambda_{mess} \tag{6}$$

where $M^u(0)$ ($M^u(1)$) denotes the Buffer class MA in position 0 (1). Th_f gives the network throughput in terms of $frame/s$, from which we computed the equivalent UDP Throughput expressed in Mbps according to the frame size considered in the experiments.

Figure 4 shows the throughput obtained using simulation and agent model as a function of the UDP network load when the packet size has an average size of 100 bytes. For network load greater then 2 Mbps the model gives a constant throughput of 2.21 Mbps while the results of the simulations are slightly higher than this value for load in the range between 2 Mbps and 16 Mbps, they become essentially identical in the range between 16 Mpbs and 20 Mbps and slightly lower for higher load. We have almost the same results for a UDP load of 1.6 Mbps obtaining a throughput of 1.595 Mbps using the simulator and of 1.584 Mbps using the MA model (the percentage absolute error is equal to 0.68%). The maximum value of percentage absolute error (17.19%) was obtained with a load of 32 Mbps when the analytical model gives a throughput higher than about 0.32 Mbps with respect to that obtained by simulator.

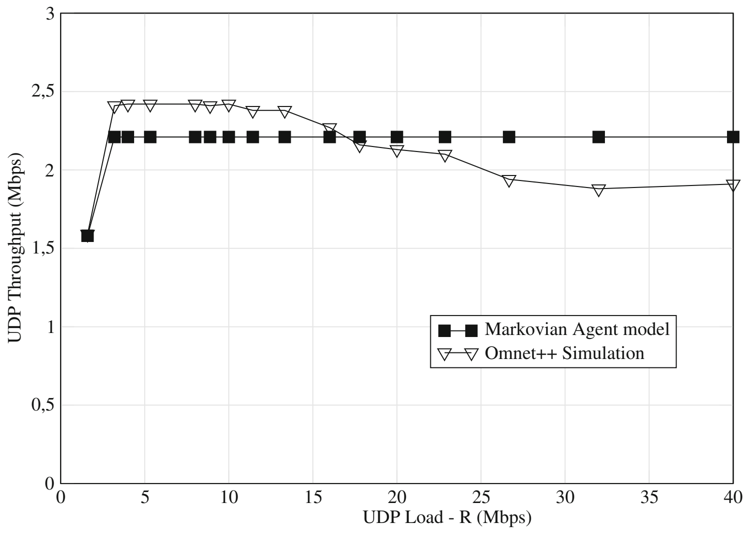

Fig. 4. UDP Throughput vs UDP Load using packets of average size equal to 100 bytes.

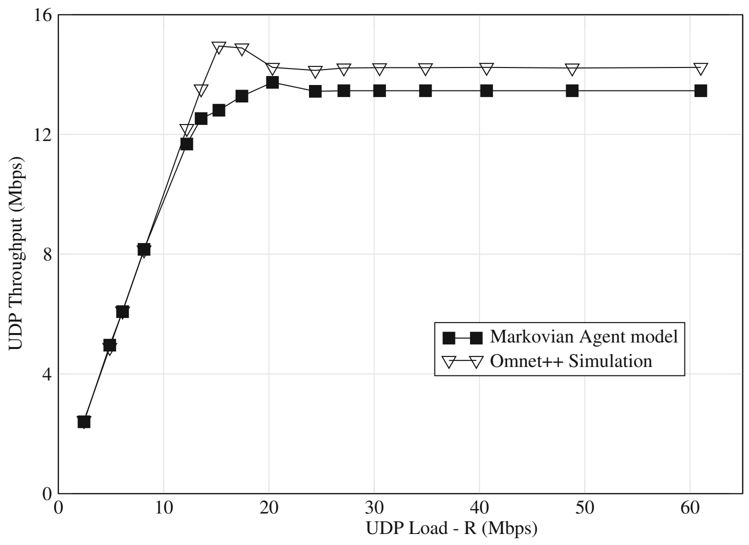

Fig. 5. UDP Throughput vs UDP Load using packets of average size equal to 1000 bytes.

Figure 5 shows the throughput obtained using simulation and MA model as a function of the UDP network load when packets have an average size of 1000 bytes. In this latter case, we obtained more accurate results. For load slower than 10 Mbps, the throughput derived from the MA model is essentially equal to that obtained with simulation (the maximum value of the percentage absolute error was 1.66%). When the load grows up over 20 Mbps the throughput saturate at 14.45 Mbps for the MA model and slightly oscillate around 14.22 Mbps for the simulations. In this latter condition the maximum value of the percentage absolute error is 5.53%. In the range between 10 Mbps and 20 Mbps the throughput obtained by the MA model presents the maximum deviation with respect to the results obtained by the simulation growing up to a percentage absolute error approximately equal to 14%.

The percentage absolute error has been computed as follows:

$$|E_{\%}| = \frac{|\mathrm{Th}_{\mathrm{model}} - \mathrm{Th}_{\mathrm{simulator}}|}{\mathrm{Th}_{\mathrm{simulator}}} \cdot 100 \qquad (7)$$

and we obtained corresponding value for comparison using linear interpolation (the actual load of the simulations is always different from the theoretical load used to obtain the throughput using the model).

It is worth to note that in the Figs. 4 and 5 we do not show the confidence intervals because they were very small: the larger confidence interval we obtained is 0.08 Mbps using a confidence of 99%; thus we do not show confidence intervals in the graphs because they are very narrow and difficult to read.

6 Conclusions and Future Work

In this paper, we introduced a new performance model of CSMA/CA based networks. Specifically Markovian agents were used to model the behavior of the protocol. In this preliminary work, we focused on a simple network scenario with 2 nodes. In the model, we neglected the presence of interference and noise and, accordingly, we set a very high SNR at the receiver in our simulations. Results obtained in terms of throughput of UDP traffic varying the network load confirmed the goodness of the approach. The results were compared with that obtained by Omnet++ simulations and using both small and large packets, experiencing very small differences.

One of the most important strengths of this proposal is the possibility to release the hypothesis of "loaded network" used in the Bianchi's based model where the collision probability is assumed constant and independent of the number of suffered retransmission. Using the proposed approach, it is instead possible to load each wireless node of the network with a different quantity of traffic and, anyway, it inherently permits to link the collision probability with the retransmission count for each wireless node without imposing a constant value.

In the future, we intend to extend this work considering a more complex scenario taking into account a greater number of wireless nodes (in order to verify model scalability), the presence of interference and/or noise (it will be possible by tuning the perception function of the messages accordingly to the distance and/or signal to noise ratio), and a multi hop (chained) scenario in which each wireless node generates its own traffic but also route traffic coming from its neighbors.

Finally, it is noteworthy to consider that the introduced Markovian agent can also be used to model wireless network using the DCF approach but different of the IEEE 802.11g. A possible candidate we intend to analyze in the future is a wireless sensor network based on the IEEE 802.15.4 protocol.

References

1. IEEE Std 802.11-2012: IEEE Standard for Information technology-telecommunications and information exchange between systems Local and metropolitan area networks-Specific requirements part 11: wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, (Revision of IEEE Std 802.11-2007), pp. 1–2793, March 2012
2. Bianchi, G.: Performance analysis of the IEEE 802.11 distributed coordination function. IEEE J. Sel. Areas Commun. **18**(3), 535–547 (2000)
3. Kadiyala, M.K., Shikha, D., Pendse, R., Jaggi, N.: Semi-Markov process based model for performance analysis of wireless LANs. In: 2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), pp. 613–618, March 2011
4. Wang, H., Kang, G., Huang, K.: An advanced Semi-Markov process model for performance analysis of wireless LANs. In: 2012 IEEE Vehicular Technology Conference (VTC Fall), pp. 1–5, September 2012

5. Prasad, Y.R.V., Pachamuthu, R.: Analytical model of adaptive CSMA-CA MAC for reliable and timely clustered wireless multi-hop communication. In: 2014 IEEE World Forum on Internet of Things (WF-IoT), pp. 212–217, March 2014
6. Felemban, E., Ekici, E.: Single hop IEEE 802.11 DCF analysis revisited: accurate modeling of channel access delay and throughput for saturated and unsaturated traffic cases. IEEE Trans. Wirel. Commun. **10**(10), 3256–3266 (2011)
7. Lin, S., Fu, L.: Unsaturated throughput analysis of physical-layer network coding based on IEEE 802.11 distributed coordination function. IEEE Trans. Wirel. Commun. **12**(11), 5544–5556 (2013)
8. Tadayon, N., Wang, H., Chen, H.H.: Performance analysis of distributed access multihop poisson networks. IEEE Trans. Veh. Technol. **63**(2), 849–858 (2014)
9. Stajkic, A., Buratti, C., Verdone, R.: Modeling multi-hop CSMA-based networks through semi-Markov chains. In: 2015 International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 520–525, August 2015
10. Gribaudo, M., Cerotti, D., Bobbio, A.: Analysis of on-off policies in sensor networks using interacting markovian agents. In: Sixth Annual IEEE International Conference on Pervasive Computing and Communications, 2008. PerCom 2008, pp. 300–305, March 2008
11. Bruneo, D., Scarpa, M., Bobbio, A., Cerotti, D., Gribaudo, M.: Markovian agent modeling swarm intelligence algorithms in wireless sensor networks. Performance Evaluation **69**(3–4), 135–149 (2012). Selected papers from ValueTools 2009. http://www.sciencedirect.com/science/article/pii/S0166531611000137