Consiglio Nazionale delle Ricerche

Performance Modelling and Measurements of TCP Transfer Throughput in 802.11-based WLANs

R. Bruno, M. Conti, E. Gregori

IIT TR-02/2006

Technical report

Maggio 2006

Istituto di Informatica e Telematica

Performance Modelling and Measurements of TCP Transfer Throughput in 802.11-based WLANs

Raffaele Bruno, Marco Conti, Enrico Gregori
Istituto di Informatica e Telematica (IIT)
Italian National Research Council (CNR)
Via G. Moruzzi, 1 - 56124 PISA, Italy
{r.bruno,m.conti,e.gregori}@iit.cnr.it

ABSTRACT

The growing popularity of the 802.11 standard for building local wireless networks has generated an extensive literature on the performance modelling of its MAC protocol. However, most of the available studies focus on the throughput analysis in saturation conditions, while very little has been done on investigating the interactions between the 802.11 MAC protocol and closed-loop transport protocols such as TCP. This paper addresses this issue by developing an analytical model to compute the stationary probability distribution of the number of backlogged nodes in a WLAN in the presence of persistent TCP-controlled download and upload data transfers. By embedding the network backlog distribution in the MAC protocol modelling, we can precisely estimate the throughput performance of TCP connections. A large set of experiments conducted in a real network validates the model correctness for a wide range of configurations. A particular emphasis is devoted to investigate and explain the TCP fairness characteristics. Our analytical model and the supporting experimental outcomes demonstrate that using default settings for the capacity of devices' output queues provides a fair allocation of channel bandwidth to the TCP connections, independently of the number of downstream and upstream flows. Furthermore, we show that the TCP total throughput does not degrade by increasing the number of wireless stations.

Keywords

802.11 MAC protocol, TCP, performance modelling, Markov chain, traffic measurements.

1. INTRODUCTION

The last years have seen an exceptional growth of the Wireless Local Area Network (WLAN) industry, with a substantial increase in the number of wireless users and applications. This growth was due, in large part, to the availability of inexpensive and highly interoperable networking solutions based on the IEEE 802.11 standards [21], and to the growing trend of providing built-in wireless network cards into mobile computing platforms. Recently, the Wi-Fi market is experiencing a renewed growth as new standardization efforts are carried out [22,35] and new market opportunities are explored with the deployment of *metro-scale* 802.11 networks, which are metropolitan areas with 802.11 coverage providing a cellular-like connectivity experience [23].

The tremendous increase in the number of wireless users, as well as the evolution from rudimentary communication services to more sophisticated and quality-sensitive applications, requires the careful design of mechanisms and policies to improve the network performance at all the layers of the protocol stack, and at the MAC layer in particular. In general, in order to design mechanisms capable of improving the network capacity and capabilities, such as efficient schedulers, adaptive resource management techniques, etc., it is fundamental to have accurate modelling tools to derive the system optimization conditions. In addition, the performance of these mechanisms are affected at various extents by the nature of the transport layer protocols adopted to deliver users' traffics, in particular for closed-loop protocols such as TCP. Thus, the operations of the transport layer protocols and their interactions with the MAC protocol should be integrated in the system modelling.

There is a considerable literature on throughput analysis for 802.11 WLANs, and the main contributions are discussed in a subsequent section of this paper. However, the majority of these studies concentrated on the performance modelling in saturation conditions [5, 11, 24, 31], and are not suitable for analysing feedback-based bidirectional transport protocols, such as TCP. Indeed, it is questionable whether saturated and unidirectional traffic models can be used to analyse feedback-based bidirectional transfers. For TCP in particular, accurate modelling requires simulating the correlation between transmissions in both directions, i.e., how data and acknowledgment packets interfere with each other. A few models [2,12,15] have relaxed the saturation assumption in developing the throughput analysis. However, these studies assumed unidirectional traffic and adopted synthetic traffic patterns, such as Poisson processes, to model the arrival rate of packets at the MAC buffer, which cannot be applied to the TCP modelling. On the other hand, there is a vast literature specifically focusing on the TCP modelling (e.g., [1,3,17,28,29] and references herein). However, these models mainly concentrate on characterizing the TCP evolution, either at the packet-based level or macroscopic level, such as to evaluate the TCP data-transfer throughputs in the presence of congestion and loss events caused by bottlenecked networks and noisy channels. These models can help to understand the impact of the network and TCP parameters on the TCP throughput, but they cannot describe the impact of the TCP flows on the underlying network, and the MAC protocol in particular. Indeed, the interaction between TCP and 802.11 MAC protocol is a fundamental point to identify and explain eventual performance issues.

Keeping in mind the aforementioned limitations, in this paper we develop a mathematical framework for evaluating the throughput performance of persistent TCP connections, which can be applied to both uplink and downlink flows. Our analysis is based on a Markovian model that is used to compute the stationary probability distribution of the buffer occupancy of both the AP's and wireless stations' transmission queues. Our model takes into account both the TCP flow-control algorithm behaviour and the MAC protocol operations. Once the buffer occupancy distributions are known, it is straightforward to derive the statistical characterization of the network backlog, i.e., the probability mass function of the number of active stations in the network. By embedding the network backlog distribution in the MAC protocol modelling, we are able to precisely estimate the throughput performance of TCP connections, independently of the network configuration. While the model validation in most of the cited papers was conducted through simulations, in this work we perform a large set of experiments on a real network to confirm the model correctness. The throughputs measured in the tests show a very good correspondence with the model predictions. In the experimentation a particular emphasis has been devoted to investigate the fairness properties of TCP traffic for a wide variety of network configurations and parameter settings. From the analytical results and the supporting experimental outcomes it turns out that: *i*) using default settings for the capacity of devices' output queues provides a fair allocation of channel bandwidth to the TCP connections, independently of the number of downstream and upstream flows; *ii*) the interaction between the contention avoidance part of the 802.11 MAC protocol and the closed-loop nature of TCP induces a bursty TCP behaviour with the TCP senders (or the TCP receivers) that could be frequently in idle state, without data (or acknowledgment) packets to transmit, while the access point stores most of the traffic generated by the TCP connections; and *iii*) the total TCP throughput does not degrade by increasing the number of wireless stations.

The results of this paper provide a new insight into the behaviour of wireless networks and an analytic model for the impact of the 802.11 MAC protocol on the transport layer protocols, and TCP in particular. The proposed model can be used to estimate the optimal protocol operating state, i.e., the state that gives the optimal performance. The optimal performance should be adopted as a reference to evaluate any enhancement to the standards. In addition, our findings can provide useful guidelines for developing more efficient and fair schedulers for the access point.

The remaining of this paper is organized as follows. Section 2 outlines related work and positions our analysis with respect to previous papers. In Section 3 we present the system model, and we discuss the modelling assumptions. Section 4 develops the throughput analysis. In Section 5 we show experimental results to validate the proposed model. Finally, Section 6 draws concluding remarks and discusses future work.

2. RELATED WORK

The literature on the IEEE 802.11 MAC protocol modelling is considerable, so that it is impossible to provide here a comprehensive overview of previous contributions. Thus, in the following we review those papers that are most related to our work.

The basic foundations for the performance modelling of the 802.11 MAC protocol can be identified in three papers [5, 11, 31]. In [5], Bianchi developed a discrete-time Markov chain model to describe the evolution of the 802.11 backoff process, assuming a constant and independent collision probability. Calí *et al.* [11] elaborated a p-persistent variant of the standard 802.11 DCF access method, assuming a constant and independent per-slot transmission probability. The persistence factor p was derived from the contention windows used in the 802.11 DCF, such that the p-persistent IEEE 802.11 protocol closely approximates the standard protocol. In [31], Tay *et al.* adopted a different modelling approach based on the *average value* analysis to derive closed-form expressions of the collision probability. These seminal models have many commonalities (e.g., the use of renewal theory arguments), and in particular they require that the system operates in *saturated conditions*, i.e., each station has always a packet to transmit. Several other papers have built on these popular models, leading to many extensions. For instance, [13] extends Bianchi's model integrating the finite retry-limit operations. Research developed in [6, 16, 38] account for additional modelling details and more precise definitions of the backoff process. The model developed in [34] is used to investigate multiple traffic classes, while [20] analyses the QoS extensions defined in the novel 802.11e standard [22]. Finally, it is worth mentioning a recent paper [24] that provides a novel fixed point formalization of the Bianchi's model, simplifying and generalizing that analysis.

The above-cited models, provided that all nodes are saturated, are very accurate in evaluating the properties of 802.11-based wireless networks. However, the saturation assumption is unlikely to be valid in more realistic configurations since most of the Internet traffic is bursty in nature and often operates in an on-off manner. For these reasons, renewed research efforts have been devoted to develop analytical tools that relax the restrictions on saturated conditions. Some modelling approaches have extended the Bianchi's analysis introducing new states – called *post-backoff* states – in the Markov chain describing the backoff process, in order to represent a node which has transmitted a packet but has none waiting [12,15]. To address the modelling difficulty, the authors in [15] assume that each station can buffer only one packet, while in [12] the MAC buffers are modelled using $M/G/1$ queues. In [2] it is proposed a model for low-rate sources, because it is assumed that each station has at most a single packet when starting a new backoff. An alternative approach is proposed in [32], in which each node is modelled as a discrete-time $G/G/1$ queue, and the average backoff window of the saturated system is used as an approximation of the one in the unsaturated case.

The review of the above-cited papers clearly points out that the performance analysis of 802.11 wireless networks under finite load sources is generally based on the use of synthetic traffic patterns, such as Poisson processes, to model the arrival rate of packets at the MAC buffer. Although this is a methodology that has been commonly used in the performance analysis of computer systems, it does not appear adequate to evaluate the performance of transport layer protocols in wireless networks. As a matter of fact, the 802.11 MAC protocol can interact with transport and application layer protocols, resulting into unexpected behaviours and performance degradations. For these reasons, recently the performance analysis of the 802.11 MAC protocol has adopted a more pragmatic approach to study the complex interactions between the various layers of the protocol stack. In particular, the analysis of TCP performance in 802.11-based networks has gathered a lot of attention. Preliminary results can be found in [24, 33], but these papers restrict the analysis to scenarios in which the nodes operate in saturated conditions. More recent contributions [9, 10, 26], have addressed the problem of accurately modelling the evolution of TCP flows in 802.11 WLANs. In [26], HTTP-like traffic is considered and the time-varying number of backlogged stations is approximated using a Bernoulli distribution. An alternative approach based on the use of discrete-time Markov chains to model the number of active stations in the networks is proposed in [9, 10], but those studies elaborate approximate analysis only for TCP receivers' advertised windows equal to one MSS. Related to our work, it is also interesting the analysis developed in [29], which investigates TCP unfairness due to AP's buffer availability. However, in our investigations we do not observe this cause of unfairness, as discussed in later sections.

3. SYSTEM MODEL

In this section we present the network and node architecture, as well as the modelling assumptions that are used throughout the analysis developed in this paper.

3.1 Network and station model

In this study, we consider a typical WLAN in which an access point (AP) provides access to external networks and Internet-based services (e.g., Web transfers, retrieval of multimedia contents, access to shared data repositories, etc.) to N^T wireless stations (in the following they are also referred to as either hosts or nodes). In this work we are not concerned with mobility issues as we are interested in the traffic dynamics of a node that shares the wireless link with a fixed set of other nodes. We assume that N^T wireless stations are communicating with a server located in the high-speed wired LAN to which the AP is connected, using TCP-controlled connections. At any instant of time, each wireless station is involved into a single upstream or downstream data flow. More precisely, we consider a configuration in which there are N_d^T TCP downstream flows (i.e., N_d^T stations are the receivers of TCP downlink connections) and N_u^T TCP upstream flows (i.e., N_u^T stations are the senders of TCP uplink connections). For the sake of brevity, henceforth we denote with S_u^T a station behaving as sender for a TCP upstream flow, and with S_d^T a station behaving as receiver for a TCP downstream flow. Note that both TCP data segments and TCP ACK packets travel over the wireless channel. The AP forwards the traffic from the server, either TCP ACK packets to the N_u^T senders or TCP data segments to the N_d^T receivers, while the S_u^T stations send TCP data segments to the AP and the S_d^T stations reply to the AP with TCP ACK packets.

To describe the subtle interactions between the IEEE 802.11 MAC protocol and the TCP protocol it is also useful to illustrate the internal node architecture, concerning the main protocol layers and the related buffer spaces. Indeed, the behaviour of the transport protocols – especially when they implement window-based flow control algorithms and congestion control techniques, as for the TCP protocol – is generally dependent on the size of transmit and receive buffers. Therefore, to get a better understanding of packet losses and buffer overflows we will briefly describe what happens when a packet, coming either from the application program or the network card (NIC), is processed by the station's operating system. This understanding will allow us to incorporate in our model the most relevant events that affect the throughput of a TCP connection. In addition, it will make easier to describe the parameter settings adopted in the experiments. During the following discussion we will refer to the diagram shown in Figure 1, which exemplifies a simplified node architecture. As shown in the figure, the *socket layer* provides the application program interface to the communication subsystem. Thus, a *TCP socket* is a socket using the TCP protocol as the communication protocol. Every TCP socket has a pair of send and receive buffers associated with it, allocated in the operating system kernel space between the application program and the network data flow. The *socket send buffer* is used to store the application data in the kernel before it is sent to the receiver by the TCP protocol. The segments stored in the send buffer are actually sent to the interface output queue – the *tx buffer* in the figure – only when the TCP sliding window algorithm allows more data to be sent. On the other hand, data received from the network is stored in the *socket receive buffer*, from where the application can read it at its own pace. This buffer allows TCP sockets to receive and process data independently of the upper application. As the application reads data, buffer space is made available to accept more input from the network. It is important to note that each TCP socket has its own buffers, but when the data goes through the TCP/IP stack it is put in a single transmission queue, which is directly attached to the NIC. Similarly, incoming packets are put by the NIC in a *backlog queue* – the *rx buffer* in the figure – from which they are processed through the TCP/IP stack and put in the correct socket receive buffer. Thus, the packet losses can occur locally in the node at the different layers and queues. Since the memory space allocated to these buffers is variable, the sizes of all these queues have a significant influence on the performance obtained by the data flows. To correctly describe the behaviour of transport protocols it is fundamental to understand when and where the buffer overflows occur. In the following section we will thoroughly elaborate on the modelling assumptions concerning the buffer capacities, while in Section 5.1 we will discuss in more depth the default queue settings usually adopted in current operating systems.

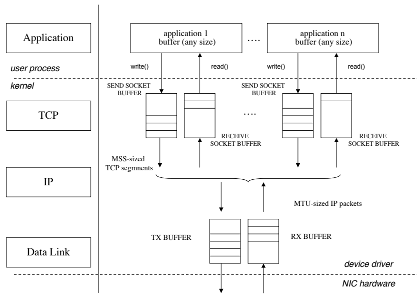


Figure 1: The station model: protocols and related buffers.

As motivated by our previous observations, the system behaviour can be modelled using a network of queues. The MAC protocol determines the order the queues (i.e., the nodes) are served and the related service time, which is the time a frame spends in the MAC layer from the instant of leaving the transmission buffer until its successful transmission. It is intuitive that frames may have different service times depending on the number of non-empty queues in the network. Indeed, as discussed in the introduction, in real 802.11 networks the saturated queues assumption is unlikely to be valid and several factors affect the average number of non-empty queues. One of these factors is the traffic arrival pattern, since most real traffic has variable demanded transmission rates with significant idle periods. In addition, the complex interaction between the upper layer protocols and the MAC protocol also plays a crucial role in determining the state of transmission buffers.

In the following section we discuss the modelling assumptions we have adopted during the analysis to make the problem mathematically tractable.

3.2 Modelling assumptions

In our analysis we made the following assumptions regarding the system.

A.1: We assume that the wireless channel is the bottlenecked link of the system. Thus, the TCP streams are limited by the rates they obtain in the WLAN. This assumption practically holds when considering typical network installations, since most wired LANs are based on 100 Mbps to 1 Gbps switched Ethernets.

A.2: We assume an ideal wireless channel that does not cause packet losses due to bit errors.

A.3: We consider only long-lived TCP connections having an infinite amount of data to send. This means that our analysis is concerned with large file transfers, such as documents or archived streaming media (e.g., from peer-to-peer-file-sharing programs).

A.4: We assume that in each wireless device the memory space allocated to the interface transmission and receive queues (i.e., the tx and rx queues in Figure 1) is large enough to avoid buffer overflows. This assumption effectively holds if the sum of the maximum TCP windows of all the connections traversing a node is less than its output queue capacity. The default values of TCP socket buffer sizes differ widely between implementations. However, with 20 TCP connections and considering typical default TCP windows, this would require a buffer of no more than a thousand packets at the AP, which is reasonable for current operating systems.

A.5: We assume that the application program at the receiver reads data from the socket receive buffer at the rate it is received form the network. Thus, the TCP ACK packets announce an advertised window equal to the total size of the TCP receive socket buffer.

A.6: The retransmission timeouts at each TCP transmitter are large enough to avoid timeouts. This assumption is commonly adopted in high-speed local networks, which are characterized by short Round Trip Times (RTTs) [28].

A.7: We assume that the delayed ACK mechanism is disabled. This implies that each TCP data segment is separately acknowledged with an immediate ACK.

Basing on the above-stated assumptions, we derive the following key observations. First of all, assumptions A.6, A.2 and A.4 imply that each TCP connection, after an initial transient phase, reaches a stationary regime in which the TCP window is maximal. Owing to assumption A.5, the upper bound for the TCP window is fixed and equal to the socket-receive buffer size. Therefore, each TCP stream is characterized by a fixed number of packets (the TCP window size), which are travelling in the network. In other words, in the steady-state condition a series of TCP data segments (in the forward direction) and TCP ACK packets (in the reverse direction) are constantly in transit from the AP to the hosts and vice versa. This observation will play a crucial role in elaborating our analysis, as clearly explained in the next section. Finally, from assumption A.1 it follows that each TCP data segment (TCP ACK packet) generated by TCP transmitters (receivers) is stored in the output queues of one of the wireless devices. Practically, this means that the AP can be considered as the end-point of the TCP connections.

Before developing the analysis, it is worth pointing out that many of these modelling assumptions are adopted in previous work [10, 24, 26] on modelling TCP connections in WLANs.

4. PERFORMANCE MODELLING OF TCP-CONTROLLED DATA TRANSFERS

The analysis is based on a Markov renewal-reward approach [19]. Following the footprints of [11], our key approximation is to assume a *p-persistent* variant of the 802.11 MAC protocol, in which the transmission probability in a randomly chosen slot time is independent from the number of frame retransmissions. However, in contrast to [11], this transmission probability p is not fixed, but it varies depending on the number of *competing* stations, i.e., the stations with non-empty transmissions queues. In the subsequent sections, we introduce the renewal cycles adopted during the analysis, and we develop the analytical tools needed to derive the stochastic relation between the transmission probability p and the number of TCP concurrent connections. Another fundamental observation relates to the methodology we used in the analysis. Whereas previous work has modelled the MAC buffers as independent queues (either $M/G/1$ or $G/G/1$ queues), in our analysis we use *aggregate* stochastic processes whenever it is possible. As it will be explained in the following, we elaborate our model to describe the evolution of the total number of TCP data segments and TCP ACK packets in the system. This information is sufficient to estimate the probability distribution of the number of backlogged stations in the network, and to discern between active TCP transmitters and active TCP receivers.

To ease the development of our model the analysis is divided into two distinct parts. First of all, we obtain the stationary probability $\pi_{x,y}$ of observing x active TCP senders and y active TCP receivers in the wireless network (a closed-form expression of the $\pi_{x,y}$ probability is presented in Section 4.1). Then, we exploit this information to express the throughput performance of TCP-controlled data transfers.

4.1 Network backlog analysis

Let us consider a fixed number N_u^T of TCP-controlled upload connections from the S_u^T stations to the AP, and a fixed number N_d^T of TCP-controlled download connections from the AP to the S_d^T stations, with $N^T = N_u^T + N_d^T$. We observe the status of the i-th TCP connection, $i \in (1, N^T)$, at a given time instant t. Let $X_{tcp}^i(t)$ be the stochastic process representing the number of TCP data segments written by the TCP socket send buffer into the TCP sender's output queue, but not yet transmitted. Similarly, let $X_{ack}^i(t)$ be the stochastic process representing the number of TCP ACK packets written by the TCP receiver in the station's output queue, but not yet transmitted to the related TCP sender. In this study we adopt the same discrete time scale as in [11]: t and $t+1$ correspond to the completion of two consecutive successful transmissions on the wireless channel. Let W denote the memory space allocated by the operating system to the TCP socket receive buffer, expressed in terms of MSS. Therefore, the maximum offered window advertised by the TCP receivers will be equal to W. Under the assumptions listed in Section 3.2, it holds that:

$$\forall t\ ,\ X_{tcp}^i(t) + X_{ack}^i(t) = W, \quad i = 1, \ldots, N^T. \tag{1}$$

It is worth pointing out that the TCP traffic is *elastic* because the TCP sending rate is regulated through a window-based flow control protocol that avoids continuous packet transmissions. In particular, two mechanisms – the flow control and the congestion control – are employed by the TCP to limit the transmission rate at the source [30]. Flow control determines the rate at which data is transmitted between senders and receivers, such as to avoid that the sender transmits more packets than the receiver can process. Congestion control defines the methods for implicitly interpreting signals from the network (e.g., received TCP acknowledgment packets or timer expirations) in order to adjust the sender's transmission rate. According to these mechanisms, wireless stations that are the receivers of TCP sessions will have feedback traffic (i.e., TCP acknowledgment packets counted by the $X_{ack}^i(t)$ process) to transmit to the AP depending on the number of TCP data packets they receive from the AP. Similarly, wireless stations that are the senders of TCP sessions will have TCP data packets (counted by the $X^i_{tcp}(t)$ process) to transmit depending on the pace of TCP acknowledgment packets received from the AP. It is evident that complex interactions regulate the joint evolution of the $X^i_{tcp}(t)$ and $X^i_{ack}(t)$ processes.

Let us consider the aggregate stochastic processes $X_{tcp}(t)=\sum_{i=1}^{N^T} X^i_{tcp}(t)$ and $X_{ack}(t)=\sum_{i=1}^{N^T} X^i_{ack}(t)$. The first step in our analysis is the modelling of the bidimensional process $\{X_{tcp}(t), X_{ack}(t)\}$, i.e., the derivation of the probability distribution functions of the number of TCP data segments and TCP ACK packets stored in the wireless network. To ease the development of the analysis we will reformulate this modelling problem into a stochastically equivalent problem. To this end, let $Y_{tcp}(t)$ be the total number of TCP data segments that are stored in the output queues of all the S^T_u stations, and $Y_{ack}(t)$ the total number of TCP ACK packets stored in the output queues of all the S^T_d stations. Furthermore, let $Z_{tcp}(t)$ and $Z_{ack}(t)$ be the total number of TCP data segments and TCP ACK packets that are stored in the AP's transmission queue, respectively. From equation (1), it follows that:

$$Y_{tcp}(t) + Z_{ack}(t) = m_u \ , \tag{2a}$$

$$Z_{tcp}(t) + Y_{ack}(t) = m_d \ , \tag{2b}$$

where $m_u = N^T_u * W$ and $m_d = N^T_d * W$, with $m = m_u + m_d$. Considering the definition of the $X_{tcp}(t)$ and $X_{ack}(t)$ processes, we can write:

$$X_{tcp}(t) = Y_{tcp}(t) + Z_{tcp}(t) \tag{3a}$$

$$X_{ack}(t) = Y_{ack}(t) + Z_{ack}(t) \ . \tag{3b}$$

From the set of equations (2) and (3), it follows that modelling the bidimensional process $\{X_{tcp}(t), X_{ack}(t)\}$ is equivalent to modelling the bidimensional process $\{Y_{tcp}(t), Y_{ack}(t)\}$. In other words, to compute the probability distribution functions of the number of TCP data segments and TCP ACK packets in all the output queues of the wireless network, it is sufficient to compute these probability distributions only for the output queues of the S^T_u and S^T_d stations.

Our aim is to compute the probability $b_{i,j}$ that in steady-state conditions there are i TCP data segments in the S^T_u stations' transmission queues and j TCP ACK packets in the S^T_d stations' transmission queues. This probability can be formally defined as follows:

$$b_{i,j} = \lim_{t\to\infty} Pr\{Y_{tcp}(t) = i, Y_{ack}(t) = j\} \ , \ i \in (0, m_u) \ , \ j \in (0, m_d) \ . \tag{4}$$

For the sake of brevity, in the following we adopt the short notation $Y_{i,j}$ to denote the system state such that $\{Y_{tcp}(t) = i, Y_{ack}(t) = j\}$. To cope with the model complexity, our key approximation is to assume that, for each transmission attempt, and regardless of the number of retransmissions suffered, the station transmission probability is a fixed value p_k, where k is the number of backlogged stations in the network. Later in this section we will describe how to express the number k of backlogged stations when the network is in state $Y_{i,j}$. To compute the sequence of the p_k values, for $k \in (1, N^T+1)$, we use the iterative algorithm firstly proposed in [11] to estimate the per-slot transmission probability p in a network with k backlogged nodes. In a real 802.11, a station transmission probability depends on the history of channel accesses, however we will show through experimentations and simulations that our approximation still provides accurate estimates of the system behaviour.

Once the p-persistence is assumed for the MAC protocol operations, it is possible to model the bidimensional process $\{Y_{tcp}(t), Y_{ack}(t)\}$ with the discrete time Markov chain depicted in Figure 2. To compute the transition probabilities of this Markov chain, it is useful to introduce some auxiliary probabilities. In particular, let us analyse what happens when there is a successful transmission on the channel and the network is in state $Y_{i,j}$. One possibility is that the AP has performed this successful transmission. In this case, let $\alpha^D_{i,j}$ ($\alpha^A_{i,j}$) be the probability that the AP delivered a TCP data segment (TCP ACK packet) stored in its transmission queue. On the other hand, it is possible that one of the wireless stations carried out this successful transmission. Consequently, let $\beta^D_{i,j}$ ($\beta^A_{i,j}$) be the probability that an S^T_u (S^T_d) station successfully transmitted a TCP data segment (TCP ACK packet) stored in its transmission queue. To express these four probabilities we must know the number of active S^T_u and S^T_d stations when the system is in state $Y_{i,j}$. To this end, we assume that the TCP data segments and the TCP ACK stored in the stations' transmission buffers are *uniformly* distributed over the stations' output queues. Note that this is a *worst-case* assumption, because it maximizes the number of active stations induced by the packet backlog. For clarity, we indicate with n^u_i the number of backlogged S^T_u stations when $Y_{tcp}(t) = i$; and with n^d_j the number of backlogged S^T_d stations when $Y_{ack}(t) = j$. From the previous assumptions, it follows that $n^u_i = \min(i, N^T_u)$ and $n^d_j = \min(j, N^T_d)$. Furthermore, let $n^a_{i,j}$ be an indicator function that has value 1 when the AP has a not-empty transmission queue, and 0 otherwise. It is intuitive to note that $n^a_{i,j} = 0$ only in the system state corresponding to $(i = m_u, j = m_d)$. Finally, exploiting standard probabilistic arguments, for $i \in (0, m_u)$, $j \in (0, m_d)$ we readily obtain that:

$$\alpha^D_{i,j} = \frac{n^a_{i,j}}{n^a_{i,j} + n^u_i + n^d_j} \cdot \frac{m_d - j}{m - i - j} \qquad \alpha^A_{i,j} = \frac{n^a_{i,j}}{n^a_{i,j} + n^u_i + n^d_j} \cdot \frac{m_u - i}{m - i - j} \tag{5a}$$

$$\beta^D_{i,j} = \frac{n^u_i}{n^a_{i,j} + n^u_i + n^d_j} \qquad \beta^A_{i,j} = \frac{n^d_j}{n^a_{i,j} + n^u_i + n^d_j} \ . \tag{5b}$$

For instance, let us consider the first formula in (5a). The first term of the product accounts for the probability that the successful transmission is carried out by the AP. The second term expresses the probability that the AP delivers a TCP data segment. In fact, in the state $Y_{i,j}$, the AP stores in its output queue $(m_d - j)$ TCP data segments and $(m_u - i)$ TCP ACK packets. The remaining formulas in the set (5) are derived using similar reasoning. By means of the set of probabilities given in formulas (5), ad adopting the short notation $Pr\{Y_{tcp}(t+1) = i_1, Y_{ack}(t+1) = j_1 | Y_{tcp}(t) = i_0, Y_{ack}(t) = j_0\} = P\{i_1, j_1 | i_0, j_0\}$, we are able to express the not null one-step transition probabilities of the Markov chain depicted in Figure 2 as follows:

$$\begin{cases} P\{i+1, j | i, j\} = \alpha^A_{i,j} & \text{if} \quad i \in (0, m_u - 1), j \in (0, m_d) \\ P\{i-1, j | i, j\} = \beta^D_{i,j} & \text{if} \quad i \in (1, m_u), j \in (0, m_d) \\ P\{i, j+1 | i, j\} = \alpha^D_{i,j} & \text{if} \quad i \in (0, m_u), j \in (0, m_d - 1) \\ P\{i, j-1 | i, j\} = \beta^A_{i,j} & \text{if} \quad i \in (0, m_u), j \in (1, m_d) \end{cases} \ . \tag{6}$$

The first equation in (6) expresses the probability that after a successful transmission the number of TCP data segments stored in the S^T_u stations' transmission buffers increases by one. This event occurs when the AP delivers a TCP ACK packet to one of the S^T_u stations. In fact, when the TCP sender receives from the TCP receiver an acknowledgement for previously sent data, it moves the *usable window* – that is how much data the TCP sender can send immediately – to the right, allowing one more data segment to be sent from the socket send buffer to the output queue. A similar reasoning applies to the third equation in (6), which expresses the probability that after a successful transmission the number of TCP ACK pack-

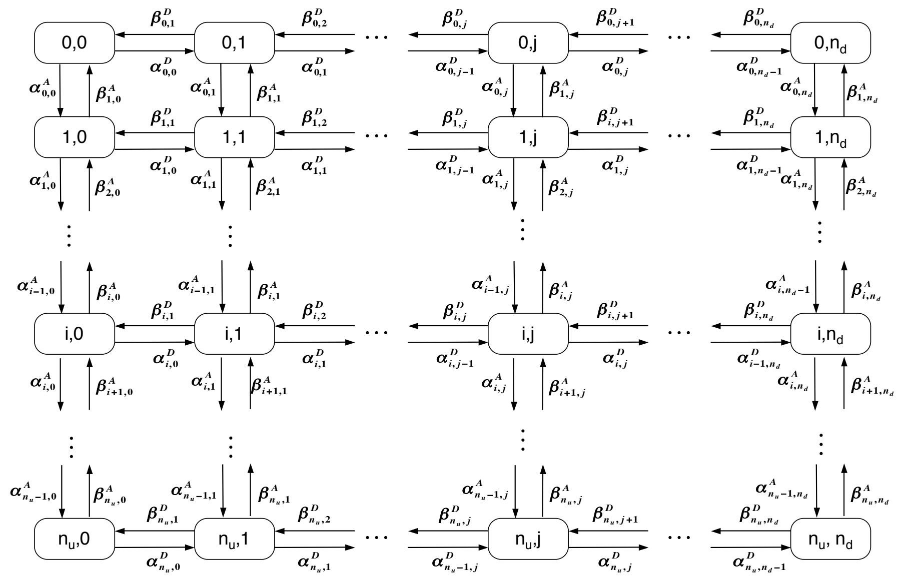


Figure 2: General Markov model of the $\{Y_{tcp}(t), Y_{ack}(t)\}$ process.

ets stored in the S_d^T stations' transmission buffers increases by one. This transition happens when the AP delivers a TCP data segment to one of the S_d^T stations. In fact, owing to the assumption A.7, each TCP data segment is immediately acknowledged. The derivation of the remaining two equations in (6) is intuitive. In particular, as considered in the second case, each successful transmission of a S_u^T station reduces by one the total number of TCP data segments stored in the S_u^T stations' output queues. Finally, the forth case models what happens when an S_d^T station successfully delivers a TCP ACK packets. When this event occurs, the total number of TCP ACK packets stored in the S_d^T stations' output queues decreases by one. Note that after each successful transmission performed by a wireless station the number of either TCP data segments or TCP ACK packets stored in the AP's output queue changes. However, this change in the AP's buffer occupancy is implicitly taken into account through the relationships (2).

Using the transition probabilities expressed in formulas (6), and showed in the Markov chain depicted in Figure 2, we can derive the relationships between the state probabilities and solve the Markov chain. Specifically, by writing the flow balance equation at the generic state $Y_{i,j}$ of this Markov chain, we obtain:

$$\begin{aligned} b_{i,j} = {} & b_{i-1,j} \cdot P\{i,j|i-1,j\} + b_{i+1,j} \cdot P\{i,j|i+1,j\} \\ & + b_{i,j-1} \cdot P\{i,j|i,j-1\} + b_{i,j+1} \cdot P\{i,j|i,j+1\} \,. \end{aligned} \tag{7}$$

By means of the flow balance equations (7) we can write a system of linear equations, which solution is the stationary distribution of the Markov chain depicted in Figure 2. This solution is uniquely determined by the normalization condition on the $b_{i,j}$ probabilities, i.e., $1 = \sum_{i=0}^{m_u} \sum_{j=0}^{m_d} b_{i,j}$. Concerning the computational complexity of solving this linear system, we can observe that the number of states, i.e., unknown variables, in the Markov chain used to model the system behaviour is equal to $l = N_u^T \cdot N_d^T \cdot W^2$. It is intuitive to note that when there are several TCP connections using large socket receive buffers, we shall solve a large-scale linear system. Therefore, it is fundamental to use optimized functions for solving our linear problem. To this end, it can be observed that in the Markov chain depicted in Figure 2 the only possible one-step transitions from a generic state are towards its adjacent states. As a consequence, the number on non-zero elements in the matrix that describes the linear system resulting from equation (7) is upper bounded by $5l$. For these reasons, we used the UMFPACK set of routines [14], which support optimized functions for solving system of linear equations in the form of $Ax = b$, when A is sparse and not symmetric. By exploiting these specialized algorithms, we have significantly limited the numerical complexity of the model, and even large-scale systems can be solved in a few seconds on a computer with a Pentium III processor at 2 GHz and 1 GB of RAM.

Once the $b_{i,j}$ probabilities are known, we can easily derive several useful probability distributions describing the system status. In particular, let Z be the total number of frames buffered in the AP's transmission queue after a generic successful transmission, and let q_z^a be the stationary probability mass function of the AP's buffer occupancy. Therefore, q_z^a can be computed as $Pr\{Z = z\}$. Given that after a successful transmission the network is in state $Y_{i,j}$, the AP stores in its output queue $(m_d - j)$ TCP data segments and $(m_u - i)$ TCP ACK packets. Thus, if follows that $\{Z|Y_{i,j}\} = m - i - j$. By exploiting this observation, the q_z^a expression ca be reformulated as:

$$q_z^a = \sum_{i=0}^{m^u} \sum_{j=0}^{m^d} Pr\{Z = z|Y_{i,j}\} \cdot b_{i,j} = \sum_{i=0}^{\min(m^u,z)} \sum_{j=0}^{m-i-z} (m-i-j) \cdot b_{i,j} \,. \tag{8}$$

In addition, we can readily compute the probability $\pi_{x,y}$ of observing x active TCP senders and y active TCP receivers at the time instant corresponding to the completion of a successful transmission. Owing to the definition of the $b_{i,j}$ steady-state probability, it immediately follows that:

$$\begin{cases} \pi_{x,y} = b_{x,y} & \text{if} \quad x < N_u^T, y < N_d^T \\ \pi_{x,y} = \sum\limits_{i=N_u^T}^{m^u} b_{i,y} & \text{if} \quad x = N_u^T, y < N_d^T \\ \pi_{x,y} = \sum\limits_{j=N_d^T}^{m^d} b_{x,j} & \text{if} \quad x < N_u^T, y = N_d^T \\ \pi_{x,y} = \sum\limits_{i=N_u^T}^{m^u} \sum\limits_{j=N_d^T}^{m^d} b_{i,j} & \text{if} \quad x = N_u^T, y = N_d^T \end{cases} \tag{9}$$

Finally, we can derive the probability distribution of the network backlog. Let K be the number of nodes, including the AP, with non-empty transmission queues after a successful transmission on the channel. Similarly, let $\hat{K}$ be the number of nodes, not including the AP, with non-empty transmission queues after a successful transmission on the channel. Let us denote with a_k ($\hat{a}_k$) the stationary probability mass function of the K ($\hat{K}$) random variable. Therefore, a_k can be computed as $Pr\{K = k\}$, while $\hat{a}_k$ is $Pr\{\hat{K} = k\}$. By definition, when the network is in state $Y_{i,j}$ it holds that $\{K|Y_{i,j}\} = n^a_{i,j} + n^u_i + n^d_j$ and $\{\hat{K}|Y_{i,j}\} = n^u_i + n^d_j$. Let Ω_k be the set of $\{i, j\}$ values such that $\{K = k|Y_{i,j}\}$ and $\widehat{\Omega}_k$ be the set of $\{i, j\}$ values such that $\{\hat{K} = k|Y_{i,j}\}$. From standard probabilistic arguments, it follows that:

$$a_k = \sum_{i,j \in \Omega_k} [n^a_{i,j} + n^u_i + n^d_j] \cdot b_{i,j} \,, \tag{10a}$$

$$\hat{a}_k = \sum_{i,j \in \widehat{\Omega}_k} [n^u_i + n^d_j] \cdot b_{i,j} \,, \tag{10b}$$

Owing to previous results, we readily obtain that $E[K] = \sum_{k=1}^{N^T+1} k \cdot a_k$ and $E[\hat{K}] = \sum_{k=0}^{N^T} k \cdot \hat{a}_k$, where $E[K]$ and $E[\hat{K}]$ denote the average number of backlogged nodes after a successful transmission, including or not including the AP, respectively.

By embedding the $b_{i,j}$ distribution in the MAC protocol modelling, we can now develop the throughput analysis.

4.2 Throughput analysis

Let ρ the overall system throughput measured at the transport layer, defined as the average number of TCP payload bits successfully transmitted per unit time. To compute ρ let us analyse what can happen between two successful transmissions, that is the time scale used in the derivation of the $b_{i,j}$ probability distribution. Hereafter, we indicate this time interval as *virtual transmission time* T_v. Furthermore, let us indicate with $T_v(i, j)$ a virtual transmission time that begins with the system in state $Y_{i,j}$. By exploiting the p-persistent assumption and renewal theoretical arguments, it is intuitive to note that the MAC protocol behaviour regenerates at the epochs corresponding to the beginning of a $T_v(0,0)$ interval. In fact, in the state $Y_{0,0}$ all the wireless stations have empty transmission queues, and the AP is the only active node in the network. From the properties of regenerative process, it follows that the time interval between two consecutive regeneration epochs – in our case the beginnings of two consecutive $T_v(0,0)$ periods – is a renewal time for the stochastic process describing the sequence of successful transmissions on the channel. Let $E[T_{renewal}]$ be the average duration of this renewal time. From the renewal-reward theory, we obtain that the ρ value is given by $E[P_{renewal}]/E[T_{renewal}]$, where $E[P_{renewal}]$ is the average number of TCP payload bits successfully transmitted during the average renewal period. The following Lemma reformulates the ρ expression, such as to make easier its mathematical derivation.

LEMMA 1. *Let $E[P_v]$ the average number of TCP payload bits transmitted during the average virtual time $E[T_v]$. By assuming the p-persistence for the MAC protocol behaviour, it holds that:*

$$\rho = \frac{E[P_v]}{E[T_v]} \,. \tag{11}$$

PROOF. Let us consider the original definition of the ρ value, that is $\rho = E[P_{renewal}]/E[T_{renewal}]$. Let Ω be the number of successful transmissions occurring during a generic renewal period. Considering the sequence of Ω virtual times forming this renewal period, we denote with T_v^j the j-th virtual time in this sequence, $j \in (1, \Omega)$, and with P_v^j the number of TCP payload bits transmitted during the T_v^j interval. Thus, it follows that $E[T_{renewal}] = E[\sum_{j=1}^{\Omega} T_v^j]$ and $E[P_{renewal}] = E[\sum_{j=1}^{\Omega} P_v^j]$. Let us indicate with $\Omega_{i,j}$, the number of $T_v(i, j)$ periods among the Ω virtual transmission times forming the $T_{renewal}$. Once independence is assumed between the virtual times, the ρ expression can be rewritten as:

$$\rho = \frac{E\left[\sum\limits_{j=1}^{\Omega} P_v^j\right]}{E\left[\sum\limits_{i=0}^{m_u}\sum\limits_{j=0}^{m_d}\sum\limits_{k=1}^{\Omega_{i,j}} T_v^k(i,j)\right]} = \frac{E[\Omega] \cdot E[P_v]}{\sum\limits_{i=0}^{m_u}\sum\limits_{j=0}^{m_d} E[\Omega(i,j)] \cdot E[T_v(i,j)]} \,. \tag{12}$$

The ratio $E[\Omega(i, j)]/E[\Omega]$ expresses the probability of being in state $Y_{i,i}$ after the completion of a successful transmission, that is the $b_{i,j}$ probability by definition. By noting that $E[T_v] = \sum_{i=0}^{m_u}\sum_{j=0}^{m_d} b_{i,j} \cdot E[T_v(i, j)]$, equation (11) immediately follows from equation (12), and this concludes the proof. $\square$

By inspecting formula (12), we observe that the throughput analysis can be divided in two distinct phases. Firstly, we have to compute the $b_{i,j}$ distribution, and this has been done in the previous section. Then, we have to compute the average duration of a virtual time given that the network is in state $Y_{i,j}$, i.e., $E[T_v(i, j)]$. Before developing the analytical derivation of the $E[T_v(i, j)]$ expression, it is useful to introduce some notations adopted during the following analysis:

- $E[R]_{i,j}$ (also $E[R(i, j)]$) is the expectation of the random variable R conditioning the network to be in state $Y_{i,j}$.
- 1_{AP} is an indicator function that has value 1 when the channel is busy and the AP is transmitting, while is 0 when the channel is busy but the AP is not transmitting.
- *SIFS*, *DIFS*, *EIFS*, and *SLOT* are the interframe spaces and the slot time used in the 802.11b MAC protocol [4], respectively; while t_B, $t_{H_{MAC}}$, and t_{ACK} are the time needed to transmit a byte, the MAC header, and the MAC acknowledgment frame at the data rate r, respectively. Finally, $t_{H_{PHY}}$ is the duration of the physical preamble.

To derive the $E[T_v(i, j)]$ expression we consider the channel events occurring during the virtual time. In particular, we can observe that before a successful transmission, collisions and *idle periods* may occur. An idle period is a time interval in which the transmission medium remains idle due to the backoff process. The key approximation in our model is that the probability, say p_k, that a station transmits in a randomly chosen slot time during the $T_v(i, j)$ period is constant. The k subscript indicates the number of backlogged stations that are active during the $T_v(i, j)$ period. Note that in the previous section we have shown that $k_{i,j} = n^a_{i,j} + n^u_i + n^d_j$ (see also the $n^a_{i,j}$, n^u_i, and n^d_j definitions in equations (5)). For brevity, we will omit the subscripts (i,j) in the k parameter. As stated previously, to compute the sequence of the p_k values, for $k \in (1, N^T+1)$, we use the iterative algorithm firstly proposed in [11] to estimate the per-slot transmission probability in a network with k backlogged nodes.

Following the footprints of [11], the average duration of the virtual time can be written as:

$$E[T_v(i,j)] = E[N_c]_{i,j} \cdot E[T_C]_{i,j} + \{E[N_c]_{i,j}+1\} \cdot E[T_I]_{i,j} + E[T_S]_{i,j} \; , \tag{13}$$

where $E[T_C]_{i,j}$, $E[T_S]_{i,j}$ and $E[T_I]_{i,j}$ are the average durations, including MAC protocol overheads, of collisions, successful transmissions and idle periods in a virtual transmission time, respectively; and $E[N_c]_{i,j}$ is the average number of collisions in a virtual time. The derivation of these unknown quantities is based on the same probabilistic arguments used in [5] and [11]. For reducing the notational complexity we define two auxiliary variables, say OV_S and OV_C, which account for the protocol overheads introduced with the *Basic Access* method [4] during a successful transmission and a collision, respectively. Let $t_H = t_{H_{MAC}} + t_{H_{PHY}}$ be the total duration of the frame header transmission, and τ be the propagation delay. Considering the packet structure and the MAC protocol operations, it follows that:

$$\begin{cases} OV_S = 2\tau + SIFS + DIFS + t_{ACK} + t_H + t_B \cdot H_1 \\ OV_C = \tau + EIFS + t_H + t_B \cdot H_1 \; , \end{cases} \tag{14}$$

where H_1 is the sum of the IP and TCP header sizes, expressed in terms of bytes.

As derived in [11], for a p-persistent MAC protocol it can be written that:

$$E[T_I]_{i,j} = \frac{(1-p_k)^k}{1-(1-p_k)^k} \cdot SLOT \; , \tag{15}$$

$$E[N_c]_{i,j} = \frac{1-(1-p_k)^k}{kp_k(1-p_k)^{k-1}} - 1 \; . \tag{16}$$

Furthermore, considering the Basic Access method, we obtain that:

$$E[T_S]_{i,j} = OV_S + E[P_S]_{i,j} \cdot t_B/8 \tag{17a}$$

$$E[T_C]_{i,j} = OV_C + E[P_C]_{i,j} \cdot t_B/8 \; , \tag{17b}$$

where $E[P_S]_{i,j}$ is the average number of TCP payload bits delivered with a successful transmission, while $E[P_C]_{i,j}$ is the average payload length of the longest packet involved in a collision event. Finally, to compute the ρ value we have to derive closed-form expressions for the $E[P_S]_{i,j}$ and $E[P_C]_{i,j}$ quantities. It is intuitive to note that these unknown quantities depend on the probability distribution of packet sizes. In this study we assume that TCP data segments have a fixed-size data content equal to l_T bits. On the other hand, since the considered TCP connections send information only in one direction, the TCP ACK packets have no useful payload. Considering that only the TCP senders generate packets with not null payloads, we can write that:

$$E[P_S]_{i,j} = E[P_S^A]_{i,j} \cdot Pr\{Succ^A|S\}_{i,j} + E[P_S^U]_{i,j} \cdot Pr\{Succ^U|S\}_{i,j} \; , \tag{18}$$

where $E[P_S^A]_{i,j}$ is the average number of TCP payload bits delivered with an AP's successful transmission, and $E[P_S^U]_{i,j}$ is the average number of TCP payload bits delivered with an S_u^T station's successful transmission; while $Pr\{Succ^A|S\}_{i,j}$ and $Pr\{Succ^U|S\}_{i,j}$ are the probabilities that observing a successful transmission, this is an AP's successful transmission or an S_u^T station's successful transmission, respectively. Owing to previous assumptions, it follows that $E[P_S^U]_{i,j} = l_T$. On the other hand, to compute $E[P_S^A]_{i,j}$, we have to consider how TCP data segments and TCP ACK packets are distributed in the AP's transmission queue. In particular, when the network is in state $Y_{i,j}$, the AP stores $(m_u - i)$ TCP ACK packets and $(m_d - j)$ TCP data segments. Hence, we readily obtain that:

$$E[P_S^A]_{i,j} = l_T \cdot \frac{m_d - j}{m - i - j} \; . \tag{19}$$

To compute the $Pr\{Succ^A|S\}_{i,j}$ and $Pr\{Succ^U|S\}_{i,j}$ probabilities we rely on the same probabilistic arguments used to derive the set of expressions in (5). In particular, by noting that when the network is in state $Y_{i,j}$ there are $(n_{i,j}^a + n_i^u + n_j^d)$ backlogged nodes, we have that:

$$Pr\{Succ^A|\}_{i,j} = \frac{n_{i,j}^a}{n_{i,j}^a + n_i^u + n_j^d} = \alpha_{i,j}^D + \alpha_{i,j}^A \tag{20a}$$

$$Pr\{Succ^U|S\}_{i,j} = \frac{n_i^u}{n_{i,j}^a + n_i^u + n_j^d} = \beta_{i,j}^D \; . \tag{20b}$$

Note that we are now also able to express the $E[P_v]$ value introduced in equation (11), by exploiting (18) and the stationary probability distribution $b_{i,j}$. Specifically, by taking the conditional expectation of $E[P_v]$ given that the network is in state $Y_{i,j}$, $E[P_v]$ can be computed as $\sum_{i=0}^{m_u}\sum_{j=0}^{m_d} b_{i,j} \cdot E[P_S]_{i,j}$. Finally, the last unknown term needed to compute the average virtual time $E[T_v]_{i,j}$ is the $E[P_C]_{i,j}$ expression (17b), that is the average length of the longest packet payload involved in a collision event. The derivation of the $E[P_C]_{i,j}$ is quite lengthy and it is reported in Appendix A.

We are now able to calculate the throughput expression (11) by noting that $E[T_v] = \sum_{i=0}^{m_u}\sum_{j=0}^{m_d} b_{i,j} \cdot E[T_v(i,j)]$. However, the throughput measure provided by (11) is an aggregate measure, whereas an important merit figure during the evaluation of the system performance will be the fairness in the TCP throughput ratio between the upstream and downstream flows. For this reason, it is useful to rewrite the ρ expression in an alternative form:

$$\rho = \rho^U + \rho^D \; , \tag{21}$$

where ρ^U is the average number of TCP payload bits successfully transmitted per unit time by the S_u^T stations, and ρ^D is the average number of TCP payload bits successfully received per unit time by the S_d^T stations. Considering Lemma 1, it is intuitive to recognize that $\rho^U = E[P_v^U]/E[T_v]$, in which $E[P_v^U]$ is the contribution to $E[P_v]$ due to the TCP payload bits transmitted by the S_u^T stations during the average virtual time $E[T_v]$. From equation (18) it follows that $E[P_v^U] = \sum_{i=0}^{m_u}\sum_{j=0}^{m_d} b_{i,j} \cdot E[P_S^U]_{i,j} \cdot Pr\{Succ^U|S\}_{i,j}$. In a similar manner, we can calculate ρ^D as the ratio $E[P_v^D]/E[T_v]$, in which $E[P_v^D]$ is the contribution to $E[P_v]$ due to the TCP payload bits received by the S_d^T stations during the average virtual time $E[T_v]$. Again, from equation (18) it follows that $E[P_v^D] = \sum_{i=0}^{m_u}\sum_{j=0}^{m_d} b_{i,j} \cdot E[P_S^A]_{i,j} \cdot Pr\{Succ^A|S\}_{i,j}$.

4.2.1 Considerations on the network backlog distribution

In this section we further elaborate on the properties of the network backlog distribution, as defined in formulas (10). In particular, we will show numerical results demonstrating the two following counter-intuitive observations: *i*) the network backlog is upper bounded to less than two stations on average; and *ii*) both the TCP receivers' advertised window sizes and the number of active TCP flows have a negligible impact on the network backlog. Note that these findings our in contrast with the behaviour of a saturated network. Indeed, it is well recognized that in saturated systems the network backlog increases by increasing the network population.

To validate our network backlog analysis we used simulations because it was not possible to monitor the driver transmission buffers in our equipments. The simulation environment is a customized tool we already adopted in our prior work [9, 10]. The parameter setting is compliant with the IEEE 802.11 DSSS physical layer, and the MAC layer configuration is summarized in Table 1. Concerning the TCP protocol we implemented the TCP Reno version [30]. The configuration of the interface buffers is also detailed in Table 1. Note that the selected capacity of the devices' output queues is large enough to ensure that no buffer overflows occur. Finally, it is worth pointing out that in Section 5 we will validate our throughput analysis in a realistic network rather than via simulations.

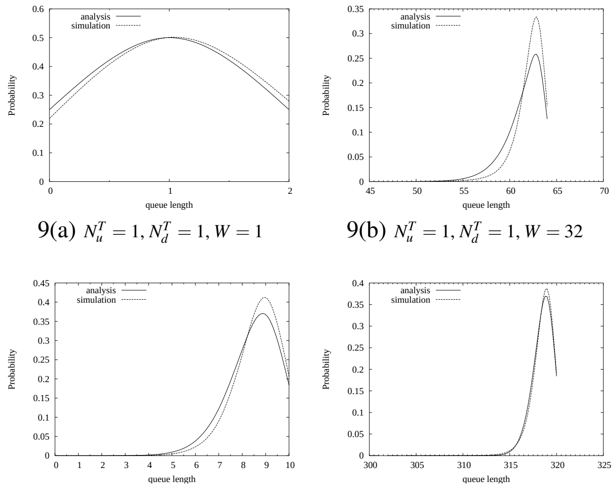
Table 1: Simulation setup.

MAC configuration			
Data rate(r)	11 Mbps	$t_{H_{PHY}}$	192 μsec
CW_{min}	32 t_{slot}	CW_{min}	1024 t_{slot}
$SIFS$	10 μsec	t_{slot}	20 μsec
$DIFS$	50 μsec	$EIFS$	364 μsec
TCP/IP Parameters			
tx queue	1000 MTUs	rx queue	1000 MTUs
H_1	52 bytes	l_T	1448 bytes

Table 2: Comparison of model predictions and observed network backlogs

W	N_u^T	N_d^T	$E[\hat{K}]$ analysis	$E[\hat{K}]$ simulation	$E[K]$ analysis	$E[K]$ simulation
1	1	1	1.00	0.92889	1.75	1.69146
1	1	2	1.30	1.17601	2.20	2.09607
1	1	5	1.49693	1.31688	2.4954	2.31605
1	1	10	1.50	1.31875	2.50	2.31875
32	1	1	1.25385	1.15385	2.25385	2.15385
32	1	2	1.39081	1.20785	2.39081	2.20785
32	1	5	1.53156	1.25558	2.52835	2.25558
32	1	10	1.50877	1.24792	2.50877	2.24792
1	2	1	1.30	1.17535	2.20	2.09562
1	5	1	1.49693	1.31627	2.4954	2.31545
1	10	1	1.50	1.31937	2.50	2.31937
32	2	1	1.39081	1.20723	2.39081	2.20723
32	5	1	1.48578	1.25685	2.48578	2.25685
32	10	1	1.49599	1.27096	2.49599	2.27096
1	2	2	1.4375	1.27527	2.40625	2.25424
1	5	5	1.50	1.31923	2.50	2.31923
1	10	10	1.50	1.31858	2.50	2.31858
32	2	2	1.45096	1.22676	2.45096	2.22676
32	5	5	1.49992	1.25388	2.49992	2.25388
32	10	10	1.50	1.24762	2.50	2.24762

The results reported in Table 2 compare the average number of backlogged stations measured through discrete-event simulations with the predictions of our model, as given by formula (10a) and (10b), for a wide range of TCP advertised window sizes and network configurations. Due to the space restrictions we will show only results related to small TCP advertised windows ($W = 1$ MSS) and large TCP advertised windows ($W = 32$ MSS), but similar behaviours have been observed in all the other cases. From the analytical outcomes, we can note that the proposed model slightly overestimates the average network backlog. This is an expected result since we developed a worst-case analysis of the network backlog. In fact, we assumed that the frames stored in the wireless network are uniformly distributed over the devices' transmissions queues, and this assumption maximizes the number of active stations induced by the packet backlog. However, this approximate description of the network backlog is sufficiently accurate to precisely estimate the TCP data-transfer throughputs, as we will show in Section 5.2.



9(a) $N_u^T = 1, N_d^T = 1, W = 1$

9(b) $N_u^T = 1, N_d^T = 1, W = 32$

9(c) $N_u^T = 5, N_d^T = 5, W = 1$

9(d) $N_u^T = 5, N_d^T = 5, W = 32$

Figure 3: Probability mass function of the AP's buffer occupancy: comparison of model predictions and simulations.

To better understand the results reported in Table 2, it is useful to investigate more in depth the AP's behaviour, in particular concerning its buffer status. To this end in Figures 3 we show the stationary probability mass function q_z^a of the AP's buffer occupancy – given by formula (8) – for some of the cases considered in the table above. The first observation we derive from the plotted curves is that there is a fairly good correspondence between the model predictions and the simulation results, which further confirms the validity of our modelling approach. In addition, we can note that it is highly probable that the AP's buffer stores almost all the frames generated by the TCP connections, and only a few frames are stored on average in the wireless stations' buffers. The reason is that the AP contends with N uplink CSMA instances. Thus, the frames are drained form the AP's buffer at a lower pace than from the wireless stations' queues, so that the AP's buffer rapidly fills up. However, the wireless stations have new frames to transmit only when the AP delivers new TCP data segments or TCP ACK packets. Thus, the closed-loop control implemented by the TCP protocol implicitly limits the network backlog.

5. EXPERIMENTAL VALIDATION OF ANALYTICAL RESULTS

To verify the accuracy of our analytical study, we conducted a series of performance tests on a real 802.11b WLAN consisting of one portable PC acting as an access point and eight portable PCs acting as wireless clients. Note that we did not use a commercial base station, but a computer instrumented as an access point to have full access to all the implementation details of the protocol stack, which are typically not made public by manufacturers.

It is evident that several factors can impact the experimental results and some of them are impossible to be isolated or controlled. For instance, wireless link interference affecting the area where the test-bed is located can have a relevant effect on the measured throughput performance. Therefore, to limit the occurrence of external interference we carried out our tests in an area of our institute not covered by other wireless networks. Furthermore, to verify our modelling hypothesis it is necessary to measure the impact of varying system parameters concerning the TCP configuration, the buffer sizes, etc. For these reasons, we decided to use a Linux kernel as operating system, such as to have access to the operating-system source code and to the implementation details of the various protocols.

In the following, first of all we detail the characteristics of our test-bed setup. After that, we compare the experimental outcomes with the analytical results obtained in Section 4. Steady-state performance figures have been estimated by replicating five times each test. The following graphs report both the averages and the 95% confidence intervals of the experimental results. Note that the confidence intervals are very tight ($\leq$ 1 percent), and are not appreciable from the plots.

5.1 Experimental setup

Our test-beds consisted of nine *IBM R50 ThinkPad* laptops equipped with an *Intel Pro-Wireless 2200* wireless card. All nodes, including the AP, use a Linux 2.6.12 kernel. All the tests are performed setting each wireless interface such as to work in IEEE 802.11b mode, transmitting at the maximal rate of 11 Mbps, with RTS/CTS disabled and using the long physical preamble. To generate the TCP traffic during the experiments we used the `iperf` tool [27]. To configure the experimental settings in accordance to our analysis, we exploit the `/proc` virtual file system that provides powerful instruments to configure several parameters of a generic TCP socket. Concerning the configuration of the interface buffers, we use the same values reported in Table 1. It is worth pointing out that the Linux kernel in version 2.6 and later defaults the capacity of each output-queue to 1000 packets. In addition, the Linux kernel in version 2.6 and later implements: *i*) the BIC TCP congestion control algorithm, *ii*) the receive buffer size auto-tuning, and *iii*) the Selective Acknowledgements (SACK) optimizations as they are defined in RFC 2883. Since our analysis takes into account the basic TCP flow-control algorithm, these additional features have been disabled. Finally, we have patched the TCP implementation to disable also the TCP delayed ack mechanism.

5.2 Performance evaluation

If not otherwise specified, the packet size is constant in all the experiments and the transport layer payload is equal to 1448 bytes. The header at the IP layer is 20 bytes long, while the header at the TCP layer is 32 bytes long, because the header contained the optional timestamp field. This corresponds to generate MTU-sized IP packets. In the following we will show results obtained using $W = 16$ MSS. We have also carried out tests investigating the impact of the size of the TCP receiver's advertised window and we obtained similar results that are not reported here for space limitations.

In Section 4.2.1 we demonstrated using simulations, that our model is capable of precisely describing the network behaviour from the perspective of the network backlog distribution. The following experimental results validate the correctness of our model to compute the TCP data transfer throughput in a large variety of network configurations. To this end, the curves plotted in Figure 4 and Figure 5 compare the throughput performance measured during the experiments with the model predictions for different mix of TCP downstream and upstream flows. More precisely, Figure 4 shows the throughput of downlink connections, while Figure 5 shows the throughput of TCP uplink connections. From the graphs, we can note that our model provides a very good correspondence between the analysis and the measured throughputs in all the considered configurations. The results plotted in Figures 4 and 5 also indicate

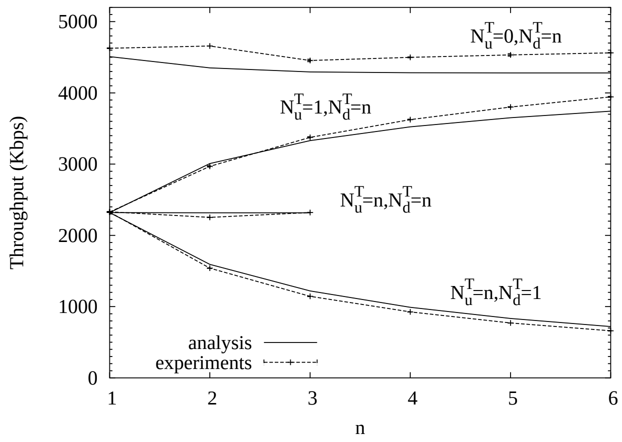


Figure 4: Comparison of model predictions with measured ρ^D.

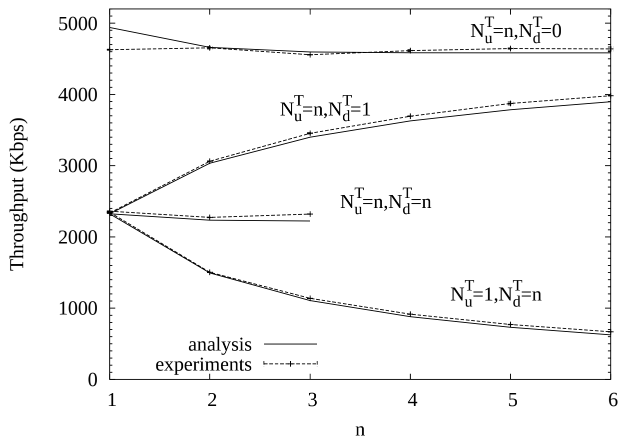


Figure 5: Comparison of model predictions with measured ρ^U.

that the TCP flows fairly share the 802.11 channel bandwidth, irrespective of the number of wireless stations that are TCP receivers or TCP senders. In addition, the aggregate TCP throughput is almost constant and independent of the number of open TCP flows. The reason is that the interactions between the MAC protocol and the TCP flow control algorithm limit the average network contention level to a few stations. Consequently, the impact of collisions on the throughput performance is negligible and independent of the number of wireless stations in the WLAN. This is in contrast with the typical behaviour of a saturated network, in which the collision probability increases by increasing the number of wireless stations. These findings further reinforce our claim that the accurate modelling of TCP in 802.11-based WLANs cannot be carried out using saturation analysis.

5.2.1 Considerations on TCP fairness

Recently, the TCP fairness in 802.11-based WLANs has been studied by several researchers and a variety of issues have been pointed out [7, 8, 18, 25, 29, 36, 37]. However, most of the performance problems revealed by these studies are rooted in the location-dependent channel capacity and errors that may characterize the wireless environment [37]. In [18] the authors observe that slow hosts may considerably limit the throughput of other hosts roughly to the level of the lower rate. In [8] it is shown that there is unfairness between long-lived and short-lived TCP flows because the short-lived TCP flows are more susceptible to losses and channel unavailability during the early stages of the TCP window growth. Unfairness between uplink and downlink TCP flows in 802.11 WLANs is observed in [7, 25, 29], but different causes are identified. In particular, these studies identify the cause of TCP unfairness in the buffer size availability at the AP. To the best of authors' knowledge the only analytical explanation of the TCP unfairness arising due to AP's buffer size availability is discussed in [29]. Possible solutions to alleviate these unfairness conditions have been also proposed. For instance, the authors in [29] propose a solution that is based on TCP receiver window manipulation at the AP, while the authors in [7,25] propose solutions based on contention window adaptation.

In our experiments we did not observe the TCP unfairness outlined in the above-cited papers, because we concentrated our investigations on different and, in some cases, more realistic network scenarios. First of all, we performed our tests in a wirelesses environment characterized by stable and homogeneous channel conditions. Thus, we did not observe asymmetry in the packet transmission rates of the different wireless stations. In addition, the packet losses are very infrequent. Hence, the congestion window growth is not hindered and the TCP timeouts are rare. Finally, the sizes of interface and driver queues are not artificially set to small values, but the default values are used. For instance, in the experiments conducted in [25] the driver buffer sizes are set to 10 packets. Similarly, simulations carried out in [29] show unfairness only for buffer capacities lower than 100 packets. On the contrary, in our experimental setup we used the system default values for the interface output queues, i.e., 1000 packets. Both our analytical and experimental results demonstrate that when there are not buffer overflows at the AP's output queue, the TCP flows fairly share the available channel bandwidth.

6. CONCLUSIONS

In this paper we investigated the throughput performance of persistent TCP-controlled download and upload data transfers in typical 802.11-based WLAN installations through analysis, simulations and experimentation. Based on a large set of experiments conducted in a real network, we found that: *i*) using default settings for the capacity of devices' output queues provides a fair throughput ratio between upload and download connections; and *ii*) the total TCP throughput does not degrade when increasing the number of wireless stations. To explain the observed TCP behaviours we developed an analytical model of the interactions between the MAC protocol and the TCP flow control algorithm. The developed analysis is used to derive the network backlog distribution and to characterize the buffers' occupancy in the presence of persistent TCP upload and download connections. By embedding the network backlog distribution in the MAC protocol modelling we were able to precisely describe the TCP evolution and to accurately evaluate the TCP throughput performance. Our analytical results show that the interaction between the contention avoidance part of the 802.11 MAC protocol and the closed-loop nature of TCP induces a bursty TCP behaviour with the TCP senders (or the TCP receivers) that could be frequently in idle state, without data (or acknowledgment) packets to transmit, while the access point stores most of the traffic generated by the TCP connections. As a consequence, the TCP traffic generates a very low contention in the WLAN.

The results of this paper provide a useful insight in the behaviour of TCP data flows in 802.11-based WLANs. In addition, our proposed model could be the analytical basis for optimizing the network capacity and improving the system capabilities. To this end we are currently exploring several possible extensions/enhancements of our modelling framework, some of which are:

- *Presence of short-lived TCP connections*: Today, several TCP sessions in Internet deliver a quite limited amount of data (e.g., the web pages carried over HTTP traffic). For short-lived TCP flows a relevant performance index is not only the obtained throughput, but also the mean delay required to complete a session. Our model could be extended to take into account realistic distributions of the TCP session durations, for instance following the modelling approach proposed in [26].

- *Interaction between TCP and UDP*: TCP traffic is elastic because it adapts its sending rate to the available link bandwidth. However, other transport protocols, such as UDP, are non-responsive because they do not reduce their sending rate in response to congestion events. As a consequence, non-responsive connections easily take advantage over TCP connections. The extension of our analysis to model the competition between responsive TCP flows and non-responsive UDP flows is part of an ongoing activity.

- *Providing higher priority to AP's transmissions*: A common idea to improve the 802.11 MAC protocol efficiency is to assign a higher priority to the AP when accessing the channel. This modification of the MAC protocol could reduce the protocol overheads, as well as make possible the implementation of more advanced traffic schedulers at the AP. We believe that our proposed model can easily provide the analytical basis to design optimized resource allocation strategies. For instance, our model can be used to compute at run-time the share of channel time to provide to the AP in order to maximize the total network capacity, or to implement user-level fairness.

7. REFERENCES

[1] A. Abouzeid, S. Roy, and M. Azizoglou. Comprehensive Performance Analysis of a TCP Session Over a Wireless Fading Link With Queueing. *IEEE Trans. Wireless Commun.*, 2(2):344–356, March 2003.

[2] F. Alizadeh-Shabdiz and S. Subramaniam. A Finite Load Analytical Model for IEEE 802.11 Distributed Coordination Fucntion MAC. In *Proc. ACM WiOpt'03*, Sophia-Antipolis, France, March 3–5 2003.

[3] E. Altman, K. Avrachenkov, and C. Barakat. A Stochastic Model of TCP/IP With Stationary Random Losses. *IEEE/ACM Trans. Networking*, 13(2):356–369, April 2005.

[4] ANSI/IEEE, New York, USA. *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification/Amendment 2: Higher-speed Physical Layer (PHY) in the* 2.4 *GHz band*, 802.11b edition, November 7 2001.

[5] G. Bianchi. Performance Analysis of the IEEE 802.11 Distributed Coordination Function. *IEEE J. Select. Areas Commun.*, 18(9):1787–1800, March 2000.

[6] G. Bianchi and I. Tinnirello. Remarks on IEEE 802.11 DCF Performance Analysis. *IEEE Commun. Lett.*, 9(8):765–767, August 2005.

[7] M. Bottigliengo, C. Casetti, C.-F. Chiasserini, and M. Meo. Smart Traffic Scheduling in 802.11 WLANs with Access Point. In *Proc. IEEE VTC 2003-Fall*, volume 4, pages 2227–2231, Orlano, FL, USA, October 6–9 2003.

[8] M. Bottigliengo, C. Casetti, C.-F. Chiasserini, and M. Meo. Short-term Fairness for TCP Flows in 802.11b WLANs. In *Proc. INFOCOM 2004*, volume 2, pages 1383–1392, Hong Kong, HK, March 7–11 2004.

[9] R. Bruno, M. Conti, and E. Gregori. Analytical Modeling of TCP Clients in Wi-Fi Hot Spot Networks. In *Proc. IFIP-TC6 Networking Conference*, pages 626–637, Athens, Greece, May 9–14 2004.

[10] R. Bruno, M. Conti, and E. Gregori. Modeling TCP Throughput over Wireless LANs. In *Proc. IMACS World Congress on Scientific Computation, Applied Mathematics and Simulation*, Paris, France, July 11–15 2005.

[11] F. Calí, M. Conti, and E. Gregori. Dynamic Tuning of the IEEE 802.11 Protocol to Achieve a Theorethical Throughput Limit. *IEEE/ACM Trans. Networking*, 8(6):785–799, 2000.

[12] G. Cantieni, Q. Ni, C. Barakat, and T. Turletti. Performance analysis of IEEE 802.11 DCF under finite load traffic. *Elsevier Computer Commun. Journal*, 28:1095–1109, June 2005.

[13] P. Chatzimisios, A. Boucouvalas, and V. Vitsas. IEEE 802.11 packet delay: a finite retry limit analysis. In *Proc. Globecom*, volume 2, pages 950–954, San Francisco, USA, December 2003.

[14] T. Davis. UMFPACK V4.4, January 28 2005. http://www.cise.ufl.edu/research/sparse/umfpack/v4.4/.

[15] K. Duffy, D. Malone, and D. Leith. Modeling the 802.11 Distributed Coordinated Function in Non-Saturated Conditions. *IEEE Commun. Lett.*, 9(8):715–717, August 2005.

[16] C. Foh and J. W. Tantra. Comments on IEE 802.11 Saturation Throughput Analysis with Freezing of Backoff Counters. *IEEE Commun. Lett.*, 9(2):130–132, February 2005.

[17] M. Garetto, R. Lo Cigno, M. Meo, and M. Marsan. Closed Queueing Network Models of Interacting Long-Lived TCP Flows. *IEEE/ACM Trans. Networking*, 12(2):300–311, April 2004.

[18] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda. Performance Anomaly of 802.11b. In *Proc. IEEE INFOCOM 2003*, volume 2, pages 836–843, San Francisco, USA, March 30– April 3 2003.

[19] D. Heyman and M. Sobel. *Stochastic Modles in Operations Research*, volume 1. McGraw-Hill, New York, NY, 2001.

[20] J. Hui and M. Devetsikiotis. A unified model for the performance analysis of IEEE 802.11e EDCA. *IEEE Trans. Commun.*, 53(9):1498–1510, September 2005.

[21] IEEE 802.11. The Working Group for WLAN Standards, April 2006. http://grouper.ieee.org/groups/802/11/.

[22] IEEE Computer society, New York, USA. *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification/Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements*, 802.11e edition, September 22 2005.

[23] R. Karrer, A. Sabharwal, and E. Knightly. Enabling Large-Scale Wireless Broadband: The Case for TAPs. *Computer Comm. Review (ACM SIGCOMM)*, 34(1):27–34, January 2004.

[24] A. Kumar, E. Altman, D. Miorandi, and M. Goyal. New Insights from a Fixed Point Analysis of Single Cell IEEE 802.11 WLANs. In *Proc. IEEE INFOCOM 2005*, volume 3, pages 1550–1561, Miami, Florida, USA, March 13–17 2005.

[25] D. Leith, P. Clifford, D. Malone, and A. Ng. TCP fairness in 802.11*e* WLANs. *IEEE Commun. Lett.*, 9(11):964–966, November 2005.

[26] D. Miorandi, A. Kherani, and E. Altman. A Queueing Model for HTTP Traffic over IEEE 802.11 WLANs. *Elsevier Computer Networks Journal*, 50(1):63–79, January 2006.

[27] NLANR/DAST. Iperf Version 2.0.2, May 3 2005. http://dast.nlanr.net/Projects/Iperf/.

[28] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP Throughput: a Simple Model and its Empirical Validation. *IEEE/ACM Trans. Networking*, 8(2):133–145, April 2000.

[29] S. Pilosof, R. Ramjee, D. Raz, Y. Shavitt, and P. Sinha. Understanding TCP fairness over Wireless LAN. In *Proc. IEEE INFOCOM 2003*, volume 2, pages 863–872, San Francisco, USA, March 30– April 3 2003.

[30] W. Stevens. *TCP/IP Illustrated, Volume 1: The Protocols*. Addison-Wesley, New York, NY, December 1993.

[31] Y. Tay and K. Chua. A Capacity Analysis for the IEEE 802.11 MAC Protocol. *Wireless Networks*, 7:159–171, 2001.

[32] O. Tickoo and B. Sikdar. A queueing model for finite load IEEE 802.11 random access. In *Proc. IEEE International Conference on Communications (ICC)*, volume 1, pages 175–179, Miami, Florida, USA, June 13–17 2004.

[33] H. Wu, Y. Peng, K. Long, S. Cheng, and J. Ma. Performance of Reliable Transport Protocol over IEEE 802.11 Wireless LAN: Analysis and Enhancement. In *Proc. IEEE INFOCOM 2002*, pages 599–607, New York, USA, June 23–27 2002.

[34] Y. Xiao. A simple and effective priority scheme for IEEE 802.11. *IEEE Commun. Lett.*, 7:70–72, February 2003.

[35] Y. Xiao. IEEE 802.11n: Enhancements for Higher Throughput in Wireless LANs. *IEEE Wireless Commun. Mag.*, 12(6):82–91, December 2005.

[36] G. Xylomenos and G. Polyzos. TCP and UDP performance over a wireless LAN. In *Proc. IEEE INFOCOM 1999*, volume 2, pages 439–446, New York, NY, USA, March 21–25 1999.

[37] G. Xylomenos, G. Polyzos, P. Mahonen, and M. Saaranen. TCP performance issues over wireless links. *IEEE Commun. Mag.*, 39(4):52–58, April 2001.

[38] E. Ziouva and T. Antonakopoulos. CSMA/CA performance under high traffic conditions: throughput and delay analysis. *Elsevier Computer Commun. Journal*, 25:313–321, February 2002.

APPENDIX

A. DERIVATION OF EXPRESSION (17b)

In this appendix we derive the $E[P_C]_{i,j}$ expression as defined in formula (17b) by assuming that each collided packet is an independent random variable P. Note that we analyse the most general case, in which we assume general probability cumulative functions $F(\cdot)$ for the discrete random variables describing the packet's payload sizes, and at the end we derive the particular cases as considered in Section 4.2. More precisely, given that the network is in state $Y_{i,j}$, let us denote with:

- P^a the payload length of an AP's transmission, and let $F^a_{i,j}(l) = Pr\{P^a \leq l\}_{i,j}$;
- P^u the payload length of an S^T_u station's transmission, and let $F^u_{i,j}(l) = Pr\{P^u \leq l\}_{i,j}$;
- P^d the payload length of an S^T_d station's transmission, and let $F^d_{i,j}(l) = Pr\{P^d \leq l\}_{i,j}$.

For brevity, we will omit the subscripts (i,j) in the probability functions. Note that P^a, P^u and P^d are discrete random variables because the payload lengths can take only integer values upper bonded by the MTU size. Let P^{max} be the maximum payload size for the IEEE 802.11b MAC protocol.

By taking the conditional expectation of $E[P_C]_{i,j}$ on the number n_c of colliding packets, $E[P_C]_{i,j}$ can be rewritten as follows

$$E[P_C]_{i,j} = \sum_{h=1}^{k} E[P_C|n_c = h]_{i,j} \cdot Pr\{n_c = h|Coll\}_{i,j} , \tag{A.1}$$

where

$$Pr\{n_c = h|Coll\}_{i,j} = \frac{\binom{k}{h} p_k^h (1-p_k)^{k-h}}{1-(1-p_k)^k - kp_k(1-p_k)^{k-1}} . \tag{A.2}$$

The $E[P_C|n_c = h]_{i,j}$ expression ca be expanded as:

$$E[P_C|n_c = h]_{i,j} = \sum_{l=0}^{P^{max}} l \cdot Pr\{P_C = l|n_c = h\} , \tag{A.3}$$

where $Pr\{P_C = l|n_c = h\}_{i,j} = Pr\{\max(P_i, P_2, \ldots, P_h) = l\}_{i,j}$, i.e., it is the probability that the maximum payload size among the h colliding frames is equal to l bits. To compute this probability it is useful to distinguish between collisions that involve or not involve the AP. Let $Pr\{1_A = 1|n_c = h\}_{i,j}$ be the probability that among the h colliding stations there is the AP, and let $Pr\{1_A = 0|n_c = h\}_{i,j}$ be the probability that among the h colliding stations there is not the AP. Then, $Pr\{P_C = l|n_c = h\}_{i,j}$ writes as follows:

$$\begin{aligned} &Pr\{P_C = l|n_c = h\}_{i,j} = \\ &\quad Pr\{P_C = l|n_c = h, 1_A = 1\}_{i,j} \cdot Pr\{1_A = 1|n_c = h\}_{i,j} + \\ &\quad Pr\{P_C = l|n_c = h, 1_A = 0\}_{i,j} \cdot Pr\{1_A = 0|n_c = h\}_{i,j} . \end{aligned} \tag{A.4}$$

To compute the probability that the AP is (is not) transmitting, given that h stations are colliding, we have to count how many ways exist to select h colliding nodes among the k active nodes in the different cases. Standard probabilistic arguments yield to:

$$Pr\{1_A = 1|n_c = h\}_{i,j} = \frac{\binom{k-1}{h-1}}{\binom{k}{h}} , \; Pr\{1_A = 0|n_c = h\}_{i,j} = \frac{\binom{k-1}{h}}{\binom{k}{h}} . \tag{A.5}$$

To derive the remaining unknown quantities in (A.4) it is useful to introduce two additional auxiliary variables, say n^u_c and n^d_c, which are the number of S^T_u and S^T_d stations participating in the collision event, respectively. For the sake of notation brevity let us denote with $A(h,s) = \{n_c = h, n^u_c = s, n^d_c = h-s-1, 1_A = 1\}$ a collision event involving the AP, a number s of S^T_u stations and a number $(h-s-1)$ of S^T_d stations; and with $B(h,s) = \{n_c = h, 1_A = 1\}$ a generic collision event involving h packets and including the AP. By noting that the probability cumulative function of the length of the longest packet payload involved in a collision with h nodes is the product of the probability cumulative functions of the colliding packet's payload sizes [11], it follows that:

$$\begin{aligned} &Pr\{P_C = l|n_c = h, 1_A = 1\}_{i,j} = \\ &\sum_{s=0}^{\min(h,n^u_i)} Pr\{P_C = l|A(h,s)\}_{i,j} \cdot Pr\{A(h,s)|B(h,s)\}_{i,j} = \\ &\sum_{s=0}^{\min(h,n^u_i)} \{\Delta(l) - \Delta(l-1)\} \cdot \frac{\binom{n^u_i}{s}\binom{n^d_j}{h-s-1}}{\binom{k-1}{h-1}} , \end{aligned} \tag{A.6}$$

where $\Delta(l) = F^a(l)[F^u(l)]^s[F^d(l)]^{h-s-1}$.

Similarly, by denoting with $A'(h,s) = \{n_c = h, n^u_c = s, n^d_c = h-s-1, 1_A = 0\}$ a collision event not including the AP, and involving a number s of S^T_u stations and a number $(h-s)$ of S^T_d stations; and with $B'(h,s) = \{n_c = h, 1_A = 1\}$ a generic collision event involving h packets and not including the AP, we can write that:

$$\begin{aligned} &Pr\{P_C = l|n_c = h, 1_A = 0\}_{i,j} = \\ &\sum_{s=0}^{\min(h,n^u_i)} Pr\{P_C = l|A'(h,s)\}_{i,j} \cdot Pr\{A'(h,s)|B'(h,s)\}_{i,j} = \\ &\sum_{s=0}^{\min(h,n^u_i)} \{\Delta'(l) - \Delta'(l-1)\} \cdot \frac{\binom{n^u_i}{s}\binom{n^d_j}{h-s}}{\binom{k-1}{h}} , \end{aligned} \tag{A.7}$$

where $\Delta'(l) = [F^u(l)]^s[F^d(l)]^{h-s}$.

By substituting (A.6), (A.7) in equation (A.4) we finally obtain $E[P_C|n_c = h]_{i,j}$.

In this study we assume that TCP data segments have a fixed-size data content equal to l_T bits and that the TCP ACK packets have no useful payload. Owing to these assumptions, it easy to recognize that $F^d(l) = u[l]$ and $F^u(l) = u[l - l_T]$, where $u[l]$ is the unit step function of the discrete random variable l, i.e., $u[l] = 1$ for $l \geq 0$, and zero otherwise. Considering that the AP stores in its output queue $(m_d - j)$ TCP data segments and $(m_u - i)$ TCP ACK packets given that the network is in the state $Y_{i,j}$, it also follows that

$$F^a_{i,j}(l) = \begin{cases} 0 & l < 0 \\ \frac{m^u - i}{m - i - j} & 0 \leq l < l_T \\ 1 & l \geq l_T \end{cases} . \tag{A.8}$$

By means of the specific relations for the $F(\cdot)$ functions in the considered case study, we can compute the $E[P_C]_{i,j}$ value as defined in (17b).