Throughput Evaluation and Enhancement of TCP Clients in Wi-Fi Hot Spots[⋆]

Raffaele Bruno, Marco Conti, and Enrico Gregori

Italian National Research Council (CNR) - IIT Institute ,
Via G. Moruzzi 1, 56126 Pisa - ITALY
{r.bruno,m.conti,e.gregori}@iit.cnr.it

Abstract. In this paper we consider a Wi-Fi *hot spot* were M users are performing TCP downloads from Internet remote servers. Our study focuses on characterizing the way the TCP flow control mechanisms affect the MAC protocol operations, and identifying the main causes of the throughput limitations shown by the TCP traffic. In particular, we show that the TCP throughput is not limited by the collision events, but by *i*) the inability of the MAC protocol to assign a higher chance of accessing the channel to the hot spot Access Point than the mobile users, and *ii*) the interaction of flow control mechanisms used at the TCP layer and the contention avoidance scheme used at the MAC layer. We propose an extension to the MAC protocol that requires only modifications of the hot spot Access Points. Our proposed enhancement allows the Access Point to send bursts of TCP packets towards the hot spot clients. We design a resource allocation protocol aimed at maximizing the success probability of the uplink transmissions by dynamically adapting the burst size to the number of users' collisions and successful transmissions. Simulations confirm the improvements of the TCP throughput achieved by our enhanced MAC protocol.

1 Introduction

Recently, the attention of manufactures and Internet providers is turning to deploying infrastructure-based wireless networks in the market of Internet public access areas, known as *hot spots*. Specifically, a hot spot can be either an area as small as a cafe and retail shop, or as large as an airport, a convention center and a hotel where people are provided with a seamless public access to the Internet. Basically, the hot spot is an area that is served by a single Wireless LAN (WLAN), or a network of WLANs where the mobile hosts access the Internet through the WLAN's Access Points (APs). Since the IEEE 802.11b technology is the dominant technology for implementing current WLANs, in this work we have considered 802.11b-based hot spots. Several researchers have devoted their efforts to investigate the performance of the 802.11 MAC protocol. Most of these

[⋆] Work carried out under the financial support of the Italian Ministry for Education and Scientific Research (MIUR) in the framework of the Projects FIRB-PERF and FIRB-VICOM.

works are analytical studies ([1–3]), which evaluate the achievable channel utilization[1] basing on the assumption of devices operating in *saturation conditions*, i.e., transmission queues never empty. This paper highlights that this assumption doesn't fit the hot spot configuration. Specifically, the majority of applications that can be envisaged for the hot spot market, are based on TCP downloads from remote servers towards the end-users via the hot spot APs (e.g., email applications, web surfing, data retrieving and so on). Therefore, the AP is the bottleneck node that affects the performances of the whole network. Furthermore, the characterization of uplink (i.e., from user to the AP) and downlink (from the AP to the users) traffic has to be different, since the AP mostly transmit TCP data packets, whereas the hot spot clients reply back with TCP ACKs.

In this paper we focus on the analysis of the MAC protocol efficiency when the hot spot AP manages M mobile/fixed users performing TCP downloads from remote servers. Due to the complexity of the problem, in this work we have performed a simulation study, and we left to a further study the analytical characterization of the system behavior. Our study is aimed at gaining a better understanding on the causes of the severe performance limitations shown by TCP traffic in the hot spot configurations. We conducted our study considering the system from the MAC protocol perspective, and we identify the way the TCP flow control mechanisms affect the MAC protocol operations. We show that the interaction of the TCP flow control mechanisms and the MAC contention avoidance scheme impedes the hot spot clients to operate in saturation conditions. Therefore most of the optimization techniques already developed to increase the MAC efficiency are not useful in the hot spot configurations, because were derived from the saturation throughput analysis (see, e.g., the discussion and references in [3]).

We have discovered that the network contention level, expressed in terms of the average number of hot spot clients that contend for the channel bandwidth slightly increase by increasing the number of active TCP flows, as it could be expected. This observation is fundamental, because it confirms that the performance limitations are not due to the contention suffered by the multiple TCP flows, but to the inability of the MAC protocol to assign a higher chance of accessing the channel to the AP than the hop spot users. In order to overcome these limitations we propose a solution based only on the modification of the MAC protocol operations in the AP without affecting the users' behavior. Specifically, our solution allows the AP to send periodically bursts of TCP data packets towards the hot spot clients by employing a null backoff to access the channel. After sending this burst of data, the AP should wait for the users' replies. We developed a theoretical analysis to compute the burst size that the AP should adopt in order to maximize the success probability of users' transmissions. We have evaluated the proposed enhancement to the MAC protocol

[1] The channel utilization is defined as the fraction of channel bandwidth used by successfully transmitted messages. Its maximum value is refereed to as protocol capacity.

via simulations. The numerical results confirm the improvements of the TCP throughput achieved by our enhanced MAC protocol.

This paper is organized as follows. In Section 2, we present the simulation results quantifying the TCP performance achievable in the considered hot spot configuration. In Section 3.1 we analytically derive the optimal AP behavior. By exploiting our analytical results, in Section 3.2 we design and evaluate our novel resource allocation protocol.

2 Study of TCP Performance in IEEE 802.11b Hot Spots

In literature measurements are already available on commercial products about the UDP [4,5] and TCP [6,7] throughput performances in 802.11 WLANs. The experimentations on test-beds are fundamental to highlight the issues of a technology, however are usually limited to observe the behavior of transport layer protocols. Furthermore, the measurements may be affected by several factors, including the link quality and the NIC's implementation details, which often preclude the possibility of conducting a rigorous study. Finally, the manufacturers don't make available to the application layer the status of relevant MAC protocol parameters, as the instantaneous backoff value, the transmission queue's occupancy, the collision events and so on. Therefore, in this paper simulations has been conducted to gather a clearer understanding of the MAC protocol operations and to identify the inefficiencies of the MAC protocol that cause the TCP performance limitations. The simulation environment we used is an extension of the one we developed in [3], which implements all the MAC and TCP protocol details. The TCP version considered is the TCP-Reno, the most worldwide adopted TCP implementation [8]. For the details on the MAC protocol overheads the reader is referred to the IEEE 802.11b specification [9].

The aims of the simulations we have conducted are: *i*) to understand the impact of multiple TCP flows on the contention level that the MAC protocol has to deal with; and *ii*) to analyze how the TCP flow control mechanisms affect the main 802.11 MAC protocol parameters, as the average backoff, the number of retransmissions, and so on. If not otherwise stated, we assume a TCP Maximum Segment Size (MSS) of 1500 bytes and a TCP advertised window size of 2^{16} bytes[2]. The AP buffer size is assumed to be 100 MSS, that in [7] has been shown to be sufficiently large to avoid undesirable TCP unfairness. Each experiment consists of 5 simulation runs, each lasting 100 seconds of simulated time. We have considered two different network setups, which we refer to as the '*TCP case*' and the '*UDP case*', respectively. In the TCP case each STA has opened an asymptotic TCP connection with the AP. This implies that the AP has always a TCP data packet to transmit to the STAs (ftp-like traffic). In the UDP case, there are M UDP flows from the AP towards the STAs and an UDP flow from each STA towards the AP. The UDP sources are CBR flows that adopt a rate such that the transmission buffers are never empty. The UDP packets generated by the

[2] This implies that the TCP advertised window size is about 43 TCP packets.

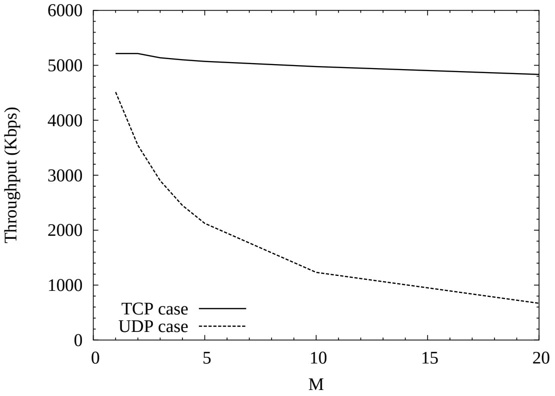

Fig. 1. The measured aggregate TCP throughput as a function of the number M of STAs

AP are 1500-bytes long packets (i.e., packets as long as the TCP data packets), and the UDP packets generated by the STAs are 40-bytes long packets (i.e., packets as long as the TCP ACK packets). The UDP case is used as *reference scenario* to quantify the impact of the TCP flow control mechanisms over the system performance. In the first set of simulations we measured the aggregate TCP throughput as a function of the hot spot population size, that is the number M of STAs. The larger is the network population, the larger is the number of devices that should contend for the channel bandwidth. According to the analysis done in other works (see, e.g., [1, 2]), the larger is the number of contending devices, the larger should be the collision probability due to the CSMA/CA access scheme and thus the lower should be the channel utilization. Surprisingly, Fig. 1 shows that the aggregate TCP throughput is slightly affected by the hot spot population size, and the TCP aggregate throughput with 20 STAs is about 93% of the TCP aggregate throughput with only one STA. To better appreciate the peculiarity of the TCP case, in Fig. 1 we have also showed the aggregate UDP throughput[3]. It is straightforward to observe that the aggregate UDP throughput significantly decreases as the number of UDP sources increases due to the higher contention level in the network. In particular the UDP throughput obtained by the AP with 20 STAs is about 15% of the UDP throughput achieved by the AP with only one STA. Finally, it is worth pointing out that the maximum channel utilization that is obtained when there is a single TCP flow is only 0.474. Neglecting the collisions and the TCP ACK traffic we can estimate that the maximum achievable throughput (see for instance the formulas derived in [6]) is 7.28 Mbps that corresponds to a channel utilization of 0.66. Hence, the target of any optimization technique that doesn't modify the 802.11 physical layer and its overheads should be to approach this theoretical limit.

[3] The aggregate TCP (UDP) throughput is defined as the sum of the throughput achieved by all the active TCP (UDP) flows.

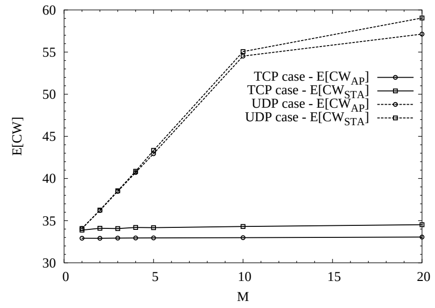

Fig. 2. Average contention window used by AP and STAs versus the number of STAs

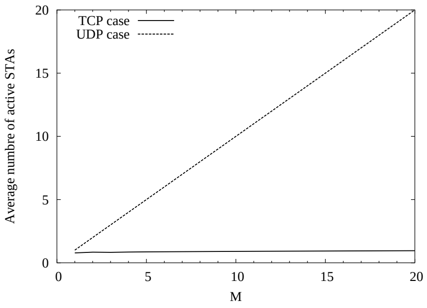

Fig. 3. Average number of active STAs after an AP's successful transmission

In the following we justify the counter-intuitive fact that the TCP throughput reduction is not due to collisions by analyzing the MAC protocol behavior. First of all, we consider the average contention windows used by the AP and the STAs, say $E[CW_{AP}]$ and $E[CW_{STA}]$ respectively. The average contention window provides a good indication of the average contention level suffered by the devices in the network. The greater is the average contention window the greater is the average time waited by the device before attempting a transmission. Fig. 2 shows the $E[CW_{AP}]$ and $E[CW_{STA}]$ values for both the TCP case and the UDP case as a function of the hot spot population. The numerical results shown in Fig. 2 highlight that in the TCP case the average contention window is slightly above 32 slots, independently of the M value, indicating that both the AP and STAs experiences a few collisions. On the other hand, for the UDP case the average contention window increases up to 59 when there are 20 STAs. This means that the devices have a significant probability to suffer at least a collision and to double the contention window using the 64-slots value. The results on the average contention windows clearly explain the difference about the throughput obtained in the TCP case and in the UDP case: TCP flows suffer a low number of collisions and there is a negligible impact of the number of TCP flows on the collision probability. This is an essential point to understand the TCP performances and in the following we provide further results to motivate this behavior.

In Fig. 3 we show the average number of STAs that after an AP's successful transmission have a packet to transmit. For the UDP case it is straightforward to observe that the number of STAs with a packet to transmit is always equal to M. On the other hand, the STAs' activity in the TCP case is strongly affected by the TCP flow control mechanisms since the amount of TCP acknowledgment traffic the STAs have to reply back to the AP depends on the amount of the TCP data traffic the AP succeeds in delivering to the STAs. Specifically, the STAs are the TCP receivers, hence they can have a new TCP ACK to transmit to the AP only after the reception of a TCP data packet from the AP. The TCP

ACK generation process is further complicated by the "Delayed ACK" technique that causes the ACK generation to be delayed for a short period of time [8]. The TCP standard also recommends that an ACK should not be delayed for more than two data packets. The TCP specification mandates that the delay must be less than 0.5 seconds, but most of the implementations use a 200 ms delay [8].

In conclusion we can state that, although the AP has always data packets to transmit, one or more STAs can be inactive, i.e., have empty transmission queues, because they have to wait to receive TCP data packets before having TCP ACKs to reply back. This behavior motivates why the average number of STAs that are active and contend for the channel bandwidth with the AP is significantly lower than M. A further relevant outcome can be driven from the results shown in Fig. 3. Specifically, the average number of active STAs, that is a measure of the average contention level in the network, slightly increases by increasing the hot spot population size, passing from 0.78 for $M = 1$ to 0.95 for $M = 20$. The explanation of this phenomenon can be found in the interaction of the TCP flow control mechanisms and the MAC contention avoidance scheme. The more traffic the AP sends to the STAs the more STAs become active. In addition, the larger is the number of active STAs the lower is the probability that the AP can experience a successful transmission. Thus the STAs tend to empty their transmissions queues and to become inactive. This interaction between the traffic sent by the AP and the traffic replied back to the AP by the STAs operates as an intrinsic closed-loop control that stabilizes the network, limiting the contention level to a few STAs on average. The results shown in Fig. 3 provide an explanation also for the fact that $E[CW_{AP}]$ is always lower than the $E[CW_{STA}]$ as indicated in Fig. 2. Specifically, it is possible that the AP transmits its packets without other STAs contending for the channel, thus avoiding collisions. On the other hand, the STAs have always at least the AP contending for the channel resources.

3 Enhancing Hot Spot Throughput

In this section we propose a simple enhancement of the 802.11 MAC protocol in order to improve the TCP throughput in the hot spot configurations considered in this work. We can identify two main concerns when proposing modifications of the MAC layer. The first one is that the modified MAC protocol could require hardware upgrades to be implemented, which is infeasible given the wide deployment of standard IEEE 802.11 NICs. Nevertheless, it is not infeasible to propose extensions to the MAC layer that involve only firmware upgrades in the network cards of the hot spot APs, without any modification in the network cards installed in the mobile hosts. The second concern is related to the compatibility requirements that any protocol extension should fulfill. It is desirable that the protocol modifications are designed in such a way that the behavior of the standard protocol is not hampered by the operations of the enhanced one. In other words, mixed scenarios should be supported where standard and modified network cards can safely inter-operate without causing performance degradations to the users owing the network cards implementing the standard protocol. The solution we propose takes into account both of these concerns as we will explain in the following.

As briefly described in Section 1, a considerable research activity has been focused to increase the MAC protocol efficiency in terms of the maximum achievable channel utilization. This goal was mainly obtained by modifying the backoff procedure in such a way to minimize the collision probability ([3, 10] and references herein). A wise choice of the contention window, which should be dynamically tuned according to the network configuration (i.e., the number of stations in the network) and traffic conditions (i.e., the distribution of the message lengths), can lead to attain significant improvements as far as the protocol capacity. However, these policies are not effective in the hot spot configurations where the causes of throughput reduction are not the collisions events, but the useless overheads that precede TCP data transmissions. Our solution is to let the AP to make its transmission attempts by using a null backoff value. This implies that the AP can start a new transmission attempt immediately after it senses the channel to be idle for a DIFS interval. This choice as a twofold remarkable result: the time required to successfully transmit a TCP packet is reduced, and the probability that an AP transmission collides with concurrent STAs' transmissions is negligible. To force the AP to use a null backoff is easy to implement because, although the binary truncated exponential backoff algorithm distribution is usually hardwired in the NIC, the distribution parameters can be set in the NIC driver. Thus, to implement a null backoff it is sufficient to set to zero the maximum contention window value in the AP's NICs. From the STAs' perspective, the 802.11 MAC protocol holds its correctness because they can continue to operate with a standard backoff.

It is worth pointing out that the optional access scheme proposed by the IEEE 802.11 standard, the *Point Coordination Function* (*PCF*) [9], is also based on the use of AP's transmissions with higher priority than STAs' transmissions. However, significant differences can be identified between our approach and the PCF. In the PCF each AP's transmission is followed by the STA's reply. When the STA has no traffic to send either to the AP or to another STA, it is mandated to deliver a null packet, further reducing the protocol efficiency. Generally, assigning a higher priority to the AP's transmissions by using a null backoff is not a sufficient condition to increase the throughput. As it will be explained in the following, differently from the PCF, we propose to separate the time intervals where the AP is allowed to deliver its traffic, from the time intervals where the STAs deliver their traffic. The duration of these time intervals will be dynamically selected in such a way to maximize the rate of STAs' successful transmissions. Basically the PCF is a polling schemes where the AP decides the order the STAs are allowed to send packets: STAs that are not polled are blocked by the AP. In our scheme the AP sends its burst of data packets almost in a contention-free manner, but it doesn't control the STAs transmissions. In fact, during the time interval reserved to the STAs' transmission, the STAs will regulate the channel access according to the standard DCF contention-based scheme.

According to our protocol, when the AP decides to perform a new transmission it seizes the channel and sends a burst of l TCP data packets. Taking into account the Delayed ACK mechanism, these l transmissions can cause at most the generation of $\lfloor l/2 \rfloor$ new TCP ACKs in the STAs. After the AP's delivery of its burst of data, the AP must let to the STAs the opportunity to transmit their queued TCP ACKs before the AP starts a new burst of transmissions. Let assume that there are m active STAs in the network, i.e., STAs with at least a TCP ACK to transmit, and that all the m STAs are using the minimum contention window, say w (Fig. 2 indicates that this assumption is a good approximation in our scenario). This assumption implies that in the next w virtual slots some of the m stations will surely perform a transmission attempt, since each STAs will uniformly select a backoff in the range $[0, \ldots, w-1]$. We use the same notation followed by Bianchi in [2] where the virtual slots can be: *i*) empty slots with duration t_{slot}, when no stations are transmitting ; *ii*) "collision" slots with duration T_c when two or more STAs collide; and *iii*) "successful" slots with duration T_s, when a single STA is transmitting. Therefore the virtual slots haven't the same *weight*. An optimal choice for the l value is the value that activates a number m of STAs such that the channel utilization during the next w virtual slots (when the STAs' transmissions are allowed) is maximal. We define as *success rate* the ratio between the number of successful slots and the time occupied by the w virtual slots, that is

$$\text{success rate} = \frac{N_s}{N_i \cdot t_{slot} + N_s \cdot T_s + N_c \cdot T_c} \,, \tag{1}$$

where N_i, N_s and N_c are the number of idle slots, successes and collisions during the w virtual slots. In order to estimate the optimal m, say m^*, we need to derive a relationship between the number of the active STAs and the success rate, such that it could be maximized. In the following section we develop an analytical framework that allows us to calculate the m^* value.

3.1 Maximizing the Success Rate

The problem we address in this section is to determine the number m of contending STAs that should be active after the AP has sent its burst of TCP data packets, in order to maximize the success rate in a window of w virtual slots[4]. To achieve our goal we need to calculate how many of these w virtual slots will be idle slots, how many will be collision slots and how many will be successful slots. Henceforth, given that there are m active STAs, we indicate the number of successful slots that will be observed during a window of w virtual slots as $E[N_s]_m^w$, the number of collisions as $E[N_c]_m^w$, and the number of idle slots as $E[N_i]_m^w$. First of all we need to express the probability, given m active STAs, that a virtual slot is a success, say $P_s(w, m)$, a collision that involves k STAs, say $P_c(w, m, k)$, or an idle slot, say $P_i(w, m)$. Following the approach used in [1],

[4] The standard MAC protocol uses an initial contention window of 32 slots, but we have carried out an analysis that is valid for a general contention window.

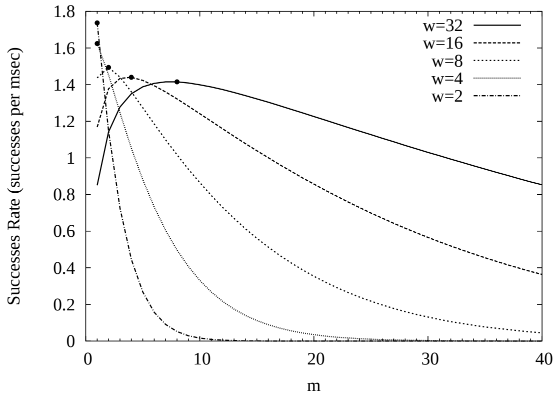

Fig. 4. Success rate as a function of m for different contention window size w

and indicating with p_w the probability that a STA is transmitting in a slot conditioned to the fact that it will try to access the channel within the following w virtual slots, we can write

$$P_s(w,m) = m \cdot p_w \cdot (1-p_w)^m \ , \tag{2a}$$

$$P_c(w,m,k) = \binom{m}{k} p_w^k \cdot (1-p_w)^{m-k} \ , \tag{2b}$$

$$P_i(w,m) = (1-p_w)^m \ . \tag{2c}$$

To derive p_w it is enough to observe that it is equiprobable that each STA tries to transmit in any of the following w virtual slots, therefore $p_w = 1/w$. By exploiting formulas (2), the following Lemma defines recursive algorithms to derive $E[N_s]_m^w$, $E[N_c]_m^w$ and $E[N_i]_m^w$.

Lemma 1. *If m active STAs uniformly try a transmission attempt during w consecutive virtual slots, the number of successful, collision and idle slots during these w virtual slots is:*

$$E[N_s]_m^w = P_s(w,m)\{1+E[N_s]_{m-1}^{w-1}\} + \sum_{k=2}^{m} P_c(w,m,k)E[N_s]_{m-k}^{w-1} + P_i(w,m)E[N_s]_m^{w-1} \ , \tag{3a}$$

$$E[N_c]_m^w = P_s(w,m)E[N_c]_{m-1}^{w-1} + \sum_{k=2}^{m} P_c(w,m,k)\{1+E[N_c]_{m-k}^{w-1}\} + P_i(w,m)E[N_c]_m^{w-1} \ , \tag{3b}$$

$$E[N_i]_m^w = P_s(w,m)E[N_i]_{m-1}^{w-1} + \sum_{k=2}^{m} P_c(w,m,k)E[N_i]_{m-k}^{w-1} + P_i(w,m) \cdot \{1+E[N_i]_m^{w-1}\} \ , \tag{3c}$$

Proof. Omitted due to the space constraints, the reader is reminded to [11].

Lemma 1 can be used to calculate the success rate as a function of the m value, hence determining the m value that maximizes it. Unfortunately, the recursive algorithm requires a considerable computational cost as m increases. To solve this problem we have also developed an efficient iterative procedure to compute formulas (3). This procedure is based on the construction of three matrixes with $(w+1)$ rows and $(m+1)$ columns, $\mathbf{S} = \{s_{i,j}\}$, $\mathbf{C} = \{c_{i,j}\}$ and $\mathbf{I} = \{i_{i,j}\}$, whose elements are defined, respectively, as: $s_{i,j} = E[N_s]^i_j$, $c_{i,j} = E[N_c]^i_j$ and $i_{i,j} = E[N_i]^i_j$, for $i = 0, 1, 2, \ldots, w$ and $j = 0, 1, 2, \ldots, m$. The matrices' elements are evaluated by exploiting the formulas derived in Lemma 1. For instance, the $s_{i,j}$ is

$$s_{i,j} = P_s(i,j) \cdot \{1 + s_{i-1,j-1}\} + \sum_{k=2}^{m} P_c(i,j,k) \cdot s_{i-1,j-k} + P_i(i,j) \cdot s_{i-1,j} \ . \quad (4)$$

Hence, the quantities defined in formulas (3) are the last element on the matrixes' diagonals. Furthermore, once we have calculate $E[N_s]^w_m$, we have for free the $E[N_s]^w_j$ for $j \leq m$, that are given by the last row of $\mathbf{S}$. Clearly, the same holds for $\mathbf{C}$ and $\mathbf{I}$. By observing (4) it is straightforward to notice that the $s_{i,j}$ element depends only on the element of the first j columns of the previous row. Therefore, to apply the iterative procedure we need only to know *a priori* the elements $\{s_{1,j}, s_{i,0}\}$ $\{c_{1,j}, c_{i,0}\}$ and $\{i_{1,j}, i_{i,0}\}$ for $i = 0, 1, 2, \ldots, w$ and $j = 0, 1, 2, \ldots, m$[5]. Let us start from the first couple. The index i indicates the size of the window where all the j STAs will try a transmission attempt. Hence, $i = 1$ implies that all the j STAs will access the channel, thus we can count a successful slot only for $j = 1$. On the other hand if $j = 0$ we cannot have transmissions. To summarize

$$s_{i,0} = 0 \quad \text{for } i = 0, 1, 2, \ldots, n \quad , \quad s_{1,j} = \begin{cases} 1 \text{ if } j = 1 \\ 0 \text{ if } j = 2, \ldots, m \end{cases} . \quad (5)$$

In the case of collisions the reasoning is clearly the opposite. In fact, if $i = 1$ we have to count a collision for $j > 1$. Hence

$$c_{i,0} = 0 \quad \text{for } i = 0, 1, 2, \ldots, w \quad , \quad c_{1,j} = \begin{cases} 0 \text{ if } j = 1 \\ 1 \text{ if } j = 2, \ldots, m \end{cases} . \quad (6)$$

The case of idle slots is different. If $i = 1$ and $j > 0$, there will be at least a transmission attempt in that slot, therefore we cannot count idle slots. If $j = 0$ we cannot have transmissions, and all the remaining i virtual slots will be idle slots. Hence

$$i_{i,0} = i \quad \text{for } i = 0, 1, 2, \ldots, w \quad , \quad i_{1,j} = 0 \quad \text{for } j = 1, 2, \ldots, n \ . \quad (7)$$

Using the initial conditions derived in formula (5), (6) and (7), we are finally able to compute the quantities defined in Lemma 1. To evaluate the duration

[5] It is straightforward to note that $s_{0,j} = c_{0,j} = i_{0,j} = 0$ for $j = 0, 1, 2, \ldots, m$.

of successful slots and collisions slots we have considered STAs sending 40-bytes long packets[6] and introduced all the MAC protocol overheads. Fig. 4 shows the success rate and the plotted curves can be exploited to easily derive the m^* value. The numerical results indicate that for all the w values analyzed the success rate is maximized for a number m^* of STAs such that $m^* = w/4$. This is a not-intuitive condition, and it was identified by using our analytical study. Further studies of this nice property are an ongoing activity beyond the scope of this paper. It is worth pointing out that this property depends on the specific setting of the MAC protocol overheads. Modifying the interframe spaces, will cause the change of the m^* value. We can observe that the m^* is lower than the m value that simply maximizes the number of successes during a window w of virtual slots. This can be explained by noting that to maximize the success rate, we try to maximize the number of successes per unit time, hence taking into account also the high cost due to collision overheads.

To summarize, if the AP operates in such a way that, after sending a burst of l TCP data packets, it has activated not more that m^* STAs, then the AP maximizes the STAs' success rate in the following contention window. To compute the m^* that maximizes the success rate we have assumed that all the STAs are using the minimum contention window w, as we have assumed that they have the same chance to try a transmission attempt during a window of w virtual time slots. However, even if the number of collision slots observed when using m^* is low (for $w = 32$ we have on average 0.77 collision slots when $m = m^*$), it cannot be neglected. Specifically, after the AP has sent a new burst of TCP data packets, in the network we will have the *"new"* STAs that has been activated by the new TCP packets, but also the *"old"* STAs that either suffered a collision in the previous contention window or didn't access the channel. To conclude, the AP has to behave in such a way that after sending a burst of l TCP data packets, in the network there are not more than m^* active STAs, but counting both newly activated STAs and previously activated STAs that didn't experience a successful transmission attempt in the previous contention window. In the following section we design and evaluate a resource allocation protocol that, exploiting our analytical results, allows the AP to increase the aggregate TCP throughput. This goal is achieved by dynamically adapting the burst size l of TCP data packets the AP sends according to the number of STAs' collisions and successful transmissions observed on the channel during a window of w virtual slots .

3.2 A Dynamic Resource Allocation Protocol for the Hot Spot AP

As said in Section 3, we propose that the AP performs its transmission attempts by using a null backoff value. This implies that the AP can start a new transmission attempt immediately after it senses the channel to be idle for a DIFS interval. The AP will send a burst of l TCP data packets, and then it will wait for the STAs replying back their TCP ACKs. How long does the AP have to wait

[6] This is the typical size of the TCP ACK.

for? The time needed to observe on the channel a number of idle slots, STAs' successful transmissions and collisions than sum up to w (where $w = 32$ to be compliant with the standard minimum contention window). Therefore the AP's behavior is cyclic: the AP's bursts of data are interleaved by 32 virtual time slots in which it doesn't participate to the channel contention. We indicate the i^{th} AP's cycle as γ_i. We show in Fig. 5 the structure of the i^{th} AP's cycle. As explained in Section 2, due to the Delayed ACK mechanism used by the TCP protocol, at most $\lfloor l/2 \rfloor$ TCP ACK can be generated in the STAs if the AP sends consecutively l TCP data packets. By assuming that before the AP burst delivery, the STAs have empty transmission queues, the number m of active STAs can be at most $\lfloor l/2 \rfloor$[7]. This assumption provides an approximation of the number of TCP ACKs in the system during an AP's cycle. The relationship between the number of TCP packets the AP has sent, and the number of STAs that have TCP ACKs to reply back is clearly more complex that the one we adopt in the following discussion. The development of a more precise characterization of the STAs' activity is an ongoing activity beyond the scope of this work. In fact, in this paper we aim at proving the effectiveness of our approach rather than to derive the optimal policy. Hence we will show that is feasible to increase the aggregate TCP throughput of the hot spot clients basing on the reduction of protocol overheads and the maximization of the uplink success rate.

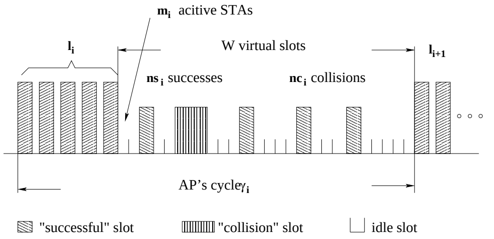

Fig. 5. Structure of channel events between during the AP's cycle γ_i .

According both to the analysis done in Section 3.1 and to the previous assumptions on the relationship between m and l, we should select the l value in such a way that $\lfloor l/2 \rfloor = m^*$ to maximize the STAs' success rate[8]. However, this choice can overload the network because the resulting scheme doesn't take into account that, due to collisions, STAs that were trying to access the channel in the γ_i AP's cycle, could contend also in the γ_{i+1} AP's cycle. Let us indicate as ns_i the number of STAs' successful transmissions occurred during γ_i, and as

[7] The number of active STAs after a burst of l TCP data packets will be exactly $\lfloor l/2 \rfloor$ only if we also assume that the AP doesn't send more that two TCP packets to the same STA.

[8] It is worth reminding that the number m^* of STAs that should be active to maximize the success rate depends only on the w value and the packet size, but it is not affected by the hot spot population size.

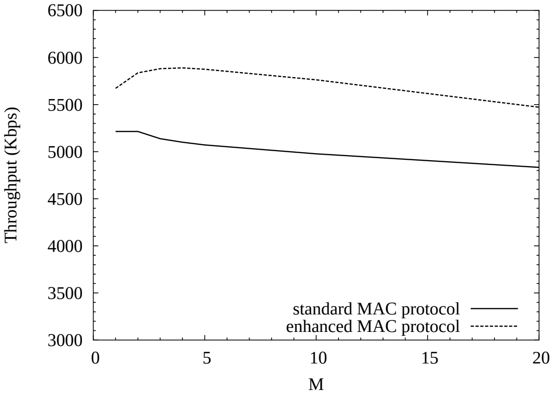

Fig. 6. The measured aggregate TCP throughput as a function of the number M of STAs

nc_i the number of STAs' collisions occurred during γ_i. To properly evaluate the number of TCP data packets the AP should send in the $(i+1)^{th}$ burst of transmissions, say l_{i+1}, we introduce a tuning factor that takes into account both the ns_i and nc_i values. The aim of this tuning is to assure that the number m_{i+1} of STAs that will try to access the channel during the γ_{i+1} is around the optimal m^* value. Taking into account all these aspects, the AP should select the burst size l_{i+1} in the following way

$$\begin{cases} l_{i+1} = 2 \cdot \left\{m^* - (\lfloor l_i/2 \rfloor - ns_i) - nc_i\right\} & \text{if } ns_i < \lfloor l_i/2 \rfloor) \ . \\ l_{i+1} = 2 \cdot \left\{m^* - nc_i\right\} & \text{otherwise,} \end{cases} \tag{8}$$

In formula (8) we differentiate the cases when the STAs perform a number of successful transmissions that is lower than the estimated number of active stations (i.e., $ns_i < \lfloor l_i/2 \rfloor$), or not. In the first case, the optimal m value is decremented not only by the number of the observed collisions during γ_i, buy also by the number of estimated STAs that have not yet transmitted their TCP ACKs.

In the following we show the numerical results obtained through simulations of an hot spot whose AP behaves accordingly to the strategy detailed in (8). In particular, we consider the same scenario used in Section 2 in order to verify the improvement achieved by adopting our resource allocation protocol. In Fig. 6 we show the aggregate TCP throughput that is achieved by the hot spot users when the AP implements either the 802.11 MAC standard protocol or employs our enhanced resource allocation protocol. We can observe that the improvement in the TCP throughput can be up to the 15%. The maximum aggregated TCP throughput we measured was 5.9 Mbps, which corresponds to a channel utilization of 0.54. As shown in Section 2, the theoretical limit obtained by ignoring collisions and TCP acknowledgments is about 0.66. Hence, a margin is still left to further improvements. It is worth pointing out that the modifications we proposed to the MAC protocol operations in the AP don't need any explicit interaction with the upper layer protocols[9]. Although the number of hot spot clients is information easily obtained by counting the number of IP addresses the AP has assigned, knowing the exact number of TCP flows that are sending traffic could be difficult. However, since the m^* value depends only on the CW_{MIN} parameter, we don't need to estimate the number of current active TCP flows. The shown results confirm that prioritizing the AP's transmissions and tuning the number of AP's transmissions to the STAs' activity level can significantly increase the aggregated TCP throughput. The design of further resource allocation policies based on more precise characterization of the STAs' activity is an ongoing research activity.

References

1. Calí, F., Conti, M., Gregori, E.: Dynamic tuning of the IEEE 802.11 protocol to achieve a theorethical throughput limit. **8** (2000) 785–799
2. Bianchi, G.: Performance Analysis of the IEEE 802.11 Distributed Coordination Function. **18** (2000) 1787–1800
3. Bruno, R., Conti, M., Gregori, E.: Optimization of Efficiency and Energy Consumption in p-Persistent CSMA-Based Wireless LANs. IEEE Trans. Mob. Comp. **1** (2002) 10–31
4. Arranz, M.G., Agüero, R., Muñoz, L., Mahõnen, P.: Behavior of UDP-Based Applications over IEEE 802.11 Wireless Networks. In: Proceedings of PIMRC'01, San Diego, CA (2001) 72–77
5. Xylomenos, G., Polyzos, G.C.: TCP and UDP Performance over a Wireless LAN. In: Proceedings of IEEE Infocom 1999, New York, NY (1999) 439–446
6. Pilosof, S., Ramjee, R., Raz, D., Shavitt, Y., Sinha, P.: Understanding TCP fairness over Wireless LAN. In: Proceedings of IEEE Infocom 2003, San Francisco, CA (2003)
7. Heusse, M., Rousseau, F., Berger-Sabbatel, G., Duda, A.: Performance Anomaly of 802.11b. In: Proceedings of IEEE Infocom 2003, San Francisco, CA (2003)
8. Stevens, W.: TCP Illustrated, Volume 1: The Protocols. Addison-Wesley, New York, NY (2001)
9. 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. ANSI/IEEE, Piscataway, NJ (2001)
10. Bononi, L., Conti, M., Donatiello, L.: Design and Performance Evaluation of a Distributed Contention Control (DCC) Mechanism for IEEE 802.11 Wireless Local Area Networks. Journal on Parallel and Distributed Computing **60** (2000)
11. Bruno, R., Conti, M., Gregori, E.: Throughput Evaluation and Enhancement of TCP Clients in Wi-Fi Hot Spots. Technical report, IIT - CNR Pisa (2003)

[9] More complex scenarios with a mix of TCP and UDP flows will require a sort of explicit interaction with the upper layers, but the investigation of this case is left to future activities.