Effective Load for Flow-Level Performance Modelling of File Transfers in Wireless LANs

G.J. Hoekstra[1,2] and R.D. van der Mei[1,3]

[1]CWI, Probability and Stochastic Networks, Amsterdam, The Netherlands
[2]Thales, Innovation Research & Technology, Huizen, The Netherlands
[3]VU University Amsterdam, Department of Mathematics, The Netherlands
{hoekstra, mei}@cwi.nl

September 23, 2010

Abstract

Today, a wide range of 802.11-based Wireless LANs (WLANs) have become dominant to provide wireless Internet access for file transfers. For engineering purposes, there is a need for very simple, explicit, yet accurate, models that predict the performance of WLANs under anticipated load conditions. In this context, several detailed packet-level models have been proposed, based on fixed-point equations. Despite the fact that these models generally lead to accurate performance predictions, they do not lead to simple explicit expressions for the performance of WLANs. Motivated by this, we propose a new analytic model that captures the highly complex combined dynamics and protocol overhead of the 802.11 MAC, IP, TCP and application-layer into an explicit expression for a single parameter which will be called the *effective service time*. Based on the effective service time, we define the *effective load* to describe the flow-level performance of file transfers over WLANs with an $M/G/1$ Processor Sharing (PS) model. Using the $M/G/1$ PS model properties we propose a simple analytic model to obtain WLAN AP buffer content distribution. Despite the fact that PS models are heavily used in modelling flow-level performance in communication networks, an extensive validation of such models has not been published in the field, or context, of WLAN. To this end, our model is validated extensively by comparing the model-based average response times against simulations. The results show that the model leads to highly accurate predictions over a wide range of parameter combinations, including light- and heavy-tailed file-size distributions and light-

and heavy-load scenarios. The simplicity and accuracy of the model make the results of high practical relevance and useful for performance engineering purposes.
keywords: Processor Sharing; wireless LANs; flow-level performance; modelling; engineering; near-insensitivity

1 Introduction

Wireless LANs are widely deployed to provide users with wireless access to private or public data networks. Pioneering work on performance models for WLANs was done by Bianchi [8], who proposed a packet-level MAC model for the saturated aggregated throughput. The work by Bianchi concerned the initial version of the IEEE 802.11 standard for WLAN MAC and Physical (PHY) layer functionality. In 1999, the IEEE 802.11 standard was ratified [1] and served as the basis for higher data-rate amendments, known as the IEEE 802.11 b/a/g/n standards [3, 2, 4, 5]. As a result of using the same protocol basis, different parameterizations of Bianchi's model appeared [30, 14, 23] to capture the saturation throughput for the higher data-rate WLAN standards. It has been observed [22, 8] that the aggregate throughput performance strongly depends on whether medium access is provided in basic access or RTS/CTS mode. In the former case, the aggregate throughput strongly decays for an increasing number of active stations, whereas in the latter case the throughput performance is far less dependent on the number of active users. *Combined* packet/flow-level models have been proposed to study the performance of non-persistent data flows with [29, 25] or without [22] using the TCP protocol. PS models have been successfully applied to model the flow-level behaviour of a variety of communication networks, including CDMA 1xEV-DO [10], WLAN-MAC [22], UMTS-HSDPA [34] and ADSL [6].

In practice, most operational WLANs do not apply any form of per-flow admission control, operate in basic access mode (having disabled RTS/CTS), and allow stations to download multiple files concurrently. For performance engineering purposes there is a great need for simple, explicit, ready-to-implement yet accurate models that predict the file transfer performance over WLANs. However, previous work on flow-level performance [29, 30, 25, 22] meets at most partially these constraints. This has motivated us to develop a new, simple, yet accurate model that *does* take into account *each* of these constraints imposed by practical deployments. A partial and preliminary version of this model appeared in [17] but has been generalized and significantly extended in the present paper. Our main interest in this paper is on the application-layer performance that is faced with the combined dynamics of all underlying protocol layers. A crucial observation is that the combined dynamics of the MAC and TCP-layer yield a remarkably simple model, compared to separate and generally complex MAC [8] and TCP-layer models.

An important observation with regard to applying PS-based models to file downloads using TCP in WLANs is made by Roijers et al. [29] and Sakurai and Hanley [30], who both state that the need to apply PS with state-dependent service rates (suggested by [10, 25] for the WLAN MAC) *vanishes* when considering TCP flows in WLAN. This is because the stations hardly contend for the medium as the WLAN AP carries most of the traffic due to its equal medium access rights. This property allows us to assume that the total available capacity in the network is constant and to subsequently obtain the *effective service time* for a given average file size. A simple, explicit model for the effective service time is formulated in Section 2. This model lays the foundation for the notion of *effective load*, which captures the protocol dynamics in a single parameter that can be used to describe the flow-level performance as a PS model.

PS models are frequently used to describe the bandwidth-sharing of TCP flows in a network. A particularly attractive feature of PS models is that they abstract from the highly complicated packet-level details of the network, but at the same time maintain the essential factors that dominate the performance, and also allow for an exact analysis. Most remarkably, despite the fact that PS models are widely used to model flow-level performance of shared media in the literature, a solid validation of PS models in the context of file transfers in WLAN in our area of interest (see below) is lacking. An important property of PS models is the well-known insensitivity property of the mean response times with respect to the file-size distribution. In an excellent survey on statistical bandwidth sharing [28], it is stated that "Even though the conditions for the insensitivity properties of this model are not realized in practice, we can be fairly confident that *actual performance does not depend significantly on detailed flow and session characteristics*, given the assumption of Poisson session arrivals".

In [25], the authors have studied HTTP throughput performance in WLANs that operate in RTS/CTS channel reservation mode with a detailed MAC model, combined with a state-dependent PS model. The proposed analytic model uses the fixed-point approach proposed by [8] and takes the TCP overhead associated to session set-up into account, specifically for HTTP applications. Users are assumed to alternate between activity periods (in which a page is downloaded) and idle periods. Under these circumstances, the number of admitted HTTP sessions in the network is limited to the number of users. Another contribution [22] proposes an integrated packet/flow-level model for TCP flows in a WLAN, assuming that a station has no more than one active TCP flow at a time (similar to [30]).

The contribution of the present paper is four-fold. First, we propose a new analytic flow-level model that translates the complex and detailed dynamics of the various protocol layers (i.e., FTP/TCP/IP/MAC), and their interactions, into an explicit expression for the 'effective service time' (denoted by β_{eff}) of the WLAN. Second, based on the effective service time we define the effective load (denoted ρ_{eff}) and use this to describe the flow-level behaviour of TCP-based file transfers over WLANs (without admission control) as an $M/G/1$ PS model with load ρ_{eff}, instead of the classical load $\rho := \lambda\beta$, where λ is the arrival rate and β is the mean service time. Third, using the $M/G/1$ PS model we propose a simple analytic model to obtain the WLAN AP buffer-content distribution. A practically useful guideline for the minimum size of the AP buffer is given for the analytic flow-level model to apply. Fourth, we provide an extensive validation of the PS model by comparing the model-based outcomes against network simulations that implement the full range of lower-layer protocol details [19]. The simulation results demonstrate that the mean response times can be accurately predicted over a wide range of parameter combinations, including light- and heavy-tailed file-size distributions and light- and heavy-load scenarios. As a by-product, the simulation results demonstrate that the mean response times and AP buffer-content distribution are *indeed* fairly (but not completely) insensitive to the file-size distribution, as suggested by the $M/G/1$ PS model, which confirms the above-mentioned statement in [28].

The remainder of this paper is organized as follows. In Section 2 we present a new analytic model for the flow-level behaviour of file transfers for various types of WLANs, explicitly taking into account the details of the protocol stack at the MAC-layer and above. In Section 3 we validate the model via network simulations, compare the outcomes against previous work [29, 30] and our proposed analytic model, observe near-insensitivity of the mean download response times to the filesize distribution, and give practical guidelines for performance engineering. Finally, topics for further research are outlined in Section 4.

2 Modelling

WLAN performance models have received much attention from the research community of which the vast majority concentrated on the Distributed Coordination Function (DCF), as specified in the IEEE 802.11 standard [1]. The most prominent analytic models are based on the Markov chain approach of Bianchi [8]. Where Bianchi's parameterization of the model concerned the Frequency-Hopping Spread Spectrum (FHSS) method, others [30, 14, 23] have applied the PHY/MAC parameters of the IEEE 802.11 b-, g- and a-standard, respectively. In more recent work [26] a detailed description is given on the use of the 802.11 a/b/e/g/n standards and a mixture thereof in the context of multi-service QoS guarantees. For a detailed description of the PHY/MAC level aspects we refer to [26], of which the 802.11 MAC parameterization details are used as a basis for the analytic model presented below.

Section 2.1 forms a brief introduction on modelling the IEEE 802.11 MAC aspects, followed by the analytic model in Section 2.2 that translates the MAC/IP/TCP-level dynamics into an explicit expression of effective service time at the application-layer. In Section 2.3 we use the latter expression to introduce the notion of effective load and use this to give an expression for the expected file transfer time.

2.1 Introduction to 802.11 overhead modelling

The Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) scheme to access the medium is used by IEEE 802.11-compliant stations (abbreviated as STA) that use the DCF. In Figure 1 a transmission cycle is shown in which two WLAN STA (an Access Point (AP) and its associated station) exchange several MAC data units, Data1-3. When a STA wants to transmit an IP packet, it is encapsulated in a MAC data unit and the STA will sense if the medium is busy or not. If the medium is idle and remains so during the following Distributed InterFrame Space (DIFS) period, the transmission may proceed. If the medium is determined to be busy, the STA waits until the end of the current transmission. If the medium is not used for a subsequent DIFS period the STA generates a random backoff period, unless the backoff timer already contains a nonzero value. Subsequently, the STA decrements its backoff timer for every time slot, τ, waited during this backoff period (indicated by $Cw1-3$). If the backoff timer reaches zero, transmission may commence.

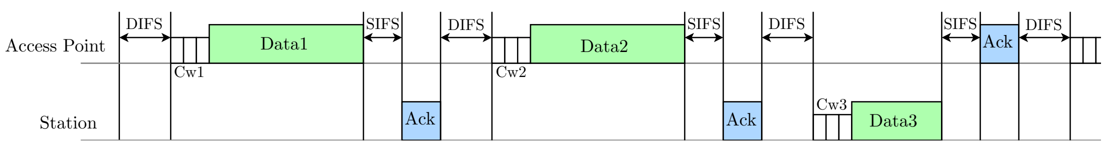


Figure 1: Transmission cycle of an IEEE 802.11-compliant station.

The random backoff time is generated from a uniform distribution, $[0, ..., Cw]$. Here the contention window parameter, Cw, is an integer within the range Cw_{min} and Cw_{max} that are defined by the various PHY standards of WLAN (cf. [3, 2, 4]). Initially, Cw is set to Cw_{min}, on every unsuccessful transmission attempt the next Cw values take the sequentially ascending integer powers of 2, minus 1, up to and including the Cw_{max} value. On every successful transmission attempt Cw is reset to Cw_{min}. On correctly receiving a MAC data unit the destination STA will send a MAC acknowledgement (ACK) to the source STA after waiting for a Short InterFrame Space (SIFS) period. Subsequently, new data transmissions

may follow after sensing the medium idle again as is illustrated in Figure 1.
If a source STA does not receive an ACK within the ACK timeout (as defined in [1]), the source station must perform a retransmission of its last MAC data unit. Retransmissions may be repeated up to the maximum retransmission threshold and cause the contention window to increase up to the value of Cw_{max} is reached. On receiving an erroneous MAC data unit (e.g. due to a collision), STAs backoff a sufficient amount of time equal to the Extended InterFrame Space (EIFS). The purpose of this backoff is to reserve enough time for a MAC ACK transmission on what was the incorrectly received frame. If a source does not receive a MAC ACK, the packet is assumed to be lost and will be retransmitted after the EIFS period. This packet retransmission follows the same procedure as on the expiration of the ACK timeout after a correct transmission.

2.2 Analytic model

We assume a network consisting of several WLAN stations and one AP, in which the stations are downloading files from an FTP server that is located close (with small propagation delay) to the AP. Each station generates FTP download requests according to an i.i.d. Poisson process and may have multiple file transfers in progress because there is no admission control mechanism on the number of file transfers per station or in total. All file transfers are carried over TCP connections that use delayed acknowledgements. Our model accounts for all overhead associated to a file download; the file transfer itself, the FTP commands and TCP handshake for opening and closing session‘s on a WLAN network operating in basic access mode, using the DCF. As explained in the previous section, STAs operating in CSMA/CA-mode that sense the medium busy must first decrement their backoff timer prior to initiating their packet transmission. Similar to [29], it is assumed in our model that a STA must perform a backoff before transmitting a TCP ACK because it finds the medium busy when transmitting its MAC ACK on the previously received TCP data segment to be acknowledged by TCP. Furthermore, the impact of collisions on the length of the backoff is not taken into account because the probability is small and the collided stations will perform their retransmissions in competition again, and thus leaving the medium idle until the first station may proceed. Finally, we assume that the handling of download requests imposes such limited CPU requirements that it can be ignored in comparison to the delay imposed by the wireless network. Practice shows that these assumptions are reasonable. Packet loss at the IP-layer is not accounted for by the model because (1) the WLAN protocol retransmits lost packets, and (2) the medium contention is low under the circumstances we consider, see the remainder of this section.

The observations and model fundamentals from [29] are used as a basis for the model proposed in this paper, but with the following extensions and modifications:

1. The flow-level model assumes the absence of admission control.
2. The model is generalized towards contemporary higher rate supplements (802.11 a/g/n).
3. The protocol overhead is modelled in more detail.
4. The backoff period modelling is refined.
5. Several MAC-level parameters are corrected.

One may suspect that the above-mentioned modifications to the model in [29] lead to a better estimation of the file download times (see Section 3 for a validation).

When considering the IEEE 802.11a/g/n standards, one can express the time, $T_d(x)$, spent by a WLAN station on transmitting a TCP *data* segment of x bits and its associated 802.11 MAC, IP and TCP overhead as:

$$T_d(x) = phy + \left\lceil \frac{mac + X_{tcp/ip} + x}{Nb} \right\rceil \cdot STT, \tag{1}$$

with $X_{tcp/ip}$ representing the TCP/IP overhead bits, mac the 802.11 MAC-layer overhead bits, SST the Symbol Transmission Time and Nb the number of coded bits in a STT of the concerned 802.11 standard. This yields a transmission rate, $R_{a/g/n} := STT/Nb$ (in bps). Note that the TCP segment and the TCP/IP/MAC overhead are transmitted in a whole number of transmission symbols. Additionally, a fixed amount of time, phy, is consumed on the medium by transmitting the overhead associated to the Physical Layer Convergence Protocol (PLCP) and by the signal extension (depending on the standard used). We refer to Table 1 for the relevant 802.11 parameters of the various IEEE 802.11 standards. After successful transmission of a data segment, the destination WLAN station will reply by transmitting a MAC *acknowledgement*. This occupies the medium for T_a seconds:

$$T_a = phy + \left\lceil \frac{ack}{Nb_c} \right\rceil \cdot STT. \tag{2}$$

where instead of the mac overhead bits a smaller ack overhead and a smaller number of coded bits per STT, Nb_c, apply. For the IEEE 802.11b standard, the following, very similar, equations apply to the time spent on transmitting the same TCP *data* segment, $T_d(x)$, and on the associated WLAN *acknowledgement*,

T_a:

$$T_d(x) = phy + \frac{mac + X_{tcp/ip} + x}{R_b}, \tag{3}$$

$$T_a = phy + \frac{ack}{R_c}, \tag{4}$$

where R_b represents the WLAN transmission rate (in bps) for data segments and R_c for the WLAN acknowledgements. When WLAN stations operate in basic access mode, source and destination stations should wait for certain inter-frame spacing times ($difs$ and $sifs$) between the transmission of WLAN MAC data and acknowledgement frames. Time $T_{da}(x)$ is defined as the time needed for MAC-acknowledged reception of a TCP segment consisting of x bits, taking into account a propagation delay of δ seconds:

$$T_{da}(x) = difs + T_d(x) + \delta + sifs + T_a + \delta. \tag{5}$$

Depending on the 802.11 standard, $T_d(x)$ and T_a may either be used from (1) and (2) or alternatively from (3) and (4). When applying TCP with delayed acknowledgements for acknowledging (the de facto) every other TCP data segment, the data is transmitted by repeated execution of a transmission cycle, encompassing the transmission of two TCP data segments by the source station and one TCP ACK segment by the destination station (depicted in Figure 1). During such a transmission cycle typically one station and the AP contend for the medium to send a TCP ACK and two TCP data segments respectively. Consequently, the AP is responsible for transmitting approximately $^2/_3$ of all packets in the network. As explained in [29, 30] the collision probability (and the total TCP throughput) is insensitive to the number of ongoing file transfers: as all flows pass through the AP, while the AP has equal MAC rights and develops a large backlog of packets in its transmission buffer. When the WLAN MAC behaviour is combined with the TCP acknowledgement mechanism, only the station that has received two TCP data segments from the AP is enabled by TCP to contend for the medium. TCP will simply inhibit all other stations (apart from those initiating a new transfer) to become active on the WLAN MAC. As a result, packet loss at the IP-layer can be neglected in our analysis because the WLAN protocol will retransmit lost packets.

In the process of contending for the medium, WLAN stations must wait a backoff time before initiating their data transmission. Still collisions may be experienced. However, in view of the observation by [29] that there is, in addition to the AP, only one station contending for the medium, we may expect that the backoff distribution is bounded by Cw_{min}. Some authors approximate the expected time consumed by the two backoff periods within a cycle by $\frac{Cw_{min}}{2}$ time slots. However, [30] specifies a more accurate analysis reasoning that for non-delayed TCP ACKs, the average backoff contribution in the file transfer transmission cycle will be the maximum of two independent observations from the uniform minimum backoff distribution $[0, ..., Cw_{min}]$. For TCP with delayed acknowledgements this means that two of the three backoff periods will contribute on average with $\frac{Cw_{min}(4Cw_{min}+5)}{6(Cw_{min}+1)}$ time slots, where it should be noted that Cw_{min} is defined, in accordance to the standard [3] and in contrast to [29], as the maximum value of the minimum backoff interval and equals 31 slots for an IEEE 802.11b PHY.

During the remaining period the AP is the only active station and will backoff, on average, $\frac{Cw_{min}}{2}$ slots after each successful transmission. As a result it can be expected that there is one backoff period per cycle in which both stations collide with probability $\frac{1}{Cw_{min}+1}$. This allows formulating the total expected time of a transmission cycle of two TCP data segments and one TCP ACK segment as:

$$\begin{aligned} T_{cycle} &= 2T_{da}(X_{MSS}) + T_{da}(X_{tcp/ip}) + \frac{Cw_{min}\,(7Cw_{min}+8)\,\tau}{6(Cw_{min}+1)} + \frac{T_{col}}{Cw_{min}+1}, \\ T_{col} &= T_d(X_{MSS}) + \delta + eifs, \end{aligned} \tag{6}$$

where T_{cycle} is the expected time of an entire transmission cycle during the file transfer, and T_{col} is the time involved in a collision on the medium. The remaining parameters are Cw_{min} (minimum contention window), τ (slot time), X_{MSS} (TCP Maximum Segment Size (MSS)), and $X_{tcp/ip}$ (TCP/IP overhead bits).

Table 1: IEEE 802.11 Parameters

Parameter	802.11a	802.11b	802.11g	802.11n
Cw_{min}(slots)	15	31	15	15
mac(bits)	246	224	246	294
τ	9μs	20μs	9μs	9μs
sifs	16μs	10μs	10μs	16μs
difs	34μs	50μs	28μs	34μs
eifs	90μs	364μs	342μs	354μs
phy	20μs	192μs	26μs	20μs
ack(bits)	134	112	134	134
δ	1μs	1μs	1μs	1μs
R_b(bps)	NA	$11 \cdot 10^6$	NA	NA
$R_{a/g/n}$(bps)	$54 \cdot 10^6$	NA	$54 \cdot 10^6$	$130 \cdot 10^6$
Nb(bits)	216	NA	216	520
R_c(bps)	NA	10^6	NA	NA
Nb_c(bits)	96	NA	96	52
STT	4μs	NA	4μs	4μs

When considering the file download response time, a certain amount of time is consumed by the file transfer. The remaining part of the traffic is exchanged for initiating and closing a TCP connection and for issuing the FTP commands and has much greater impact if file transfers become shorter and are hardly considered in WLAN flow-level performance models. TCP connection initiation involves a 3-way handshake of TCP (SYN) segments and for closing the sessions a 4-way handshake (FIN, ACK) is used [31]. In the interest of simplicity, the FTP application is modelled to use one TCP session for the FTP commands and the file transfer.

During the TCP set-up cycle, a station starts by initiating a TCP SYN segment, which is followed by a SYN ACK segment by the AP and is concluded by a TCP ACK from the station. The AP will acknowledge on the WLAN medium the packet carrying the TCP SYN from the station. As the AP has always packets to transmit, the SYN ACK is highly likely to be transmitted after and before a pair of TCP data segments from other flows.

When a station is transmitting the first (TCP SYN) and third (TCP ACK) segment a collision may occur with a small or large packet, assumed with equal probability. The expected time of a collision during these two sequences are $(T_{shortcol} + T_{col})/2$. The second (SYN ACK) segment may collide only with a TCP ACK and thus contributes $T_{shortcol}$ to the expected collision time during TCP set-up, which explains the last term in (7). The total expected time spent in backoff by the AP after transmitting its SYN ACK is assumed equal to the minimum of the backoff windows drawn by the AP and a station, corresponding to the expectation of the minimum of two uniformly i.i.d. observations from the backoff interval, thus contributing with an expected delay of $\frac{Cw_{min}(2Cw_{min}+1)}{6(Cw_{min}+1)}$ time slots. The total time spent on average in a TCP set-up then equals:

$$\begin{aligned} T_{tcp\_setup} =& 3T_{da}(X_{tcp/ip}) + \frac{Cw_{min}(2Cw_{min}+1)\tau}{6(Cw_{min}+1)} + \frac{(2T_{shortcol} + T_{col})}{Cw_{min}+1}, \\ T_{shortcol} =& T_d(X_{tcp/ip}) + \delta + eifs. \end{aligned} \tag{7}$$

As soon as the TCP connection is established, the station issues the FTP GET command. It is assumed that the FTP GET command (assumed equally sized as trivial ftp (tftp) requests) has a size of 512 bytes or 4096 bits(X_{FTP}). Since the station's backoff time does not inhibit the AP from using the medium it is not explicitly modelled, the average time spent on using the medium for the FTP GET request equals:

$$T_{FTPget} = T_{da}(X_{FTPget}) + \frac{T_{col}}{Cw_{min}+1}. \tag{8}$$

Note that the TCP ACK on the FTP GET request is not modelled here because the first TCP data segment of the file transfer will piggyback the TCP ACK and is already accounted for in the file transfer transmission cycle. As both the AP and the station rival for the medium, the collision probability is included in the above expression. The file transfer is concluded by the transmission of the last data segment, which is immediately followed by an FTP close command with an assumed size of 8 bytes ($X_{FTPclose}$). The expected size of the last data segment of the file (for non-deterministic file-size distributions) approximately equals $\frac{X_{MSS}}{2}$, and hence:

$$\begin{aligned} T_{lastcycle} =& T_{da}(X_{FTPclose}) + T_{da}\Big(\frac{X_{MSS}}{2}\Big) + T_{da}(X_{tcp/ip}) + \frac{T_{halfMSSCol}}{Cw_{min}+1} \\ &+ \frac{Cw_{min}\,(7Cw_{min}+8)\,\tau}{6(Cw_{min}+1)} \\ &+ \frac{1}{2}\Big(T_{da}(X_{tcp/ip}) + T_{shortCol} + \frac{Cw_{min}(2Cw_{min}+1)\tau}{6(Cw_{min}+1)}\Big), \\ T_{halfMSSCol} =& T_d\Big(\frac{X_{MSS}}{2}\Big) + \delta + eifs. \end{aligned} \tag{9}$$

After sending the last TCP data segment (FTP close command), the AP will contend with the station that attempts to transmit its last TCP ACK and later sending its TCP FIN. As an even number is considered equally likely as an odd number of data segments per transfer, the additional overhead related to sending an additional TCP ACK is accounted for this proportion accordingly. As a result of the connection closure, two cycles need to follow in which the station and the AP contend for the medium and concurrently decrement their backoff timer. Possible collisions ($T_{shortcol}$) will be shorter as the most likely involved segments are of size $X_{tcp/ip}$. The expected time to close the TCP connection can then be expressed as:

$$T_{TCP\_close} = 4T_{da}(X_{tcp/ip}) + \frac{2Cw_{min}(4Cw_{min}+5)\tau}{6(Cw_{min}+1)} + \frac{2T_{shortcol}}{Cw_{min}+1}, \tag{10}$$

Note that the TCP closure was not accounted for in the analysis of [25] because the closure of the TCP connection does not affect the download response time, however, it needs to be noted that the TCP set-up, closure and FTP commands do contribute to the contention and overall load on the network and to a lesser degree to an overestimation of individual download response times. The expected time consumed by the FTP commands and the TCP session opening/closing as defined by (7), (8) and (10), can be expressed as:

$$T_{tcpftp_{OH}} = T_{tcp\_setup} + T_{FTP\_get} + T_{TCP\_close}. \tag{11}$$

Now, we obtain the effective service time, β_{eff}, of the file transfer as observed at the application-layer by combining (6), (9), and (11):

$$\beta_{eff} = \frac{\Big(X_{file} - \frac{X_{MSS}}{2}\Big)T_{cycle}}{2X_{MSS}} + T_{lastcycle} + T_{tcpftp_{OH}}, \tag{12}$$

with X_{file} as the mean file size (in bits). Note that in our modelling approach, we have assumed the use of TCP with delayed acknowledgements. Although this meets many practical circumstances, some TCP implementations acknowledge each individual data segment (instead of every two segments). To account for this effect, our model may be adapted to the use of non-delayed acknowledgements by replacing T_{cycle}, $T_{lastcycle}$ and β_{eff} from (6),(9) and (12) by $\hat{T}_{cycle}$, $\hat{T}_{lastcycle}$ and $\hat{\beta}_{eff}$, respectively, defined as follows:

$$\hat{T}_{cycle} = \quad T_{da}(X_{MSS}) + T_{da}(X_{tcp/ip}) + \frac{Cw_{min}\,(4Cw_{min}+5)\,\tau}{6(Cw_{min}+1)} + \frac{T_{col}}{Cw_{min}+1}, \tag{13}$$

$$\begin{aligned}\hat{T}_{lastcycle} =&T_{da}(X_{FTPclose}) + T_{da}\left(\frac{X_{MSS}}{2}\right) + 2T_{da}(X_{tcp/ip}) + Cw_{min}\tau \\ &+ \frac{T_{halfMSSCol} + T_{shortCol}}{Cw_{min}+1},\end{aligned} \tag{14}$$

$$\hat{\beta}_{eff} = \frac{\left(X_{file} - \frac{X_{MSS}}{2}\right)\hat{T}_{cycle}}{X_{MSS}} + \hat{T}_{lastcycle} + T_{tcpftp_{OH}}. \tag{15}$$

Now the transmission cycle in (13) comprises the transmission of one TCP data segment followed by one TCP acknowledgement. Consequently, the average back-off contribution now corresponds to the two periods (identified in the paragraph preceding (6)) and does not affect the expression for the collisions because the AP was not competing with the station for the second packet transmission that is now eliminated from the transmission cycle. For the last transmission cycle, the last data segment (with expected size $\frac{X_{MSS}}{2}$) is now always acknowledged separately from the FTP close command. The resulting expected backoff contribution is obtained by combining the one from (13) and the backoff associated to the TCP ACK transmission from (9) that was used for an even number of data segments.

2.3 Effective load

To model the flow-level behaviour of file transfers, we consider a classical $M/G/1$ PS model, with flow-arrival rate λ, and where the service time B is generally distributed with mean β. In this model, incoming flows immediately enter the system, thereby receiving a fair share of the available capacity. Then the occupation rate is $\rho := \lambda\beta$, and the expected sojourn time is known to be $E[S] = \beta/(1-\rho)$. Note here that the sojourn time is *insensitive* to the service-time distribution, i.e., depends on the service-time distribution only through its mean β.

To translate the analytic model for WLAN file downloads (discussed above) into an $M/G/1$ PS model, we define the following notion of *effective load*:

$$\rho_{eff} := \lambda \cdot \beta_{eff}, \tag{16}$$

where β_{eff} is the 'effective service time' defined in (12). The quantity ρ_{eff} can be viewed as the effective medium utilization resulting from the load introduced by processing λ file download requests per second. Since the file download in the WLAN network encompasses the file transfer and the introduced overhead of FTP and TCP, the expected file transfer time (denoted $E[R]$) is modelled as the expected sojourn time in an $M/G/1$ PS model with load ρ_{eff}:

$$E[R] = \frac{\beta_{eff}}{1 - \rho_{eff}}. \tag{17}$$

Thus, to apply the analytic model, the average file download time $E[R]$ is obtained from (17), where ρ_{eff} is given by (16), and β_{eff} follows from (12). Note that β_{eff} in (16) and (17) should be replaced by $\hat{\beta}_{eff}$ from (15) when using non-delayed TCP acknowledgements instead of delayed acknowledgements.

In [21] it is shown that the steady-state distribution of N, the number of jobs in an M/G/1 PS system (defined earlier), is given by, for $n = 0, 1, \ldots$,

$$P\left(N = n\right) = \left(1 - \rho\right) \rho^{n}, \tag{18}$$

independent of the service-time distribution (for given β). In the context of file transfers in WLAN, we view the number of customers in an $M/G/1$ PS system as the number of file downloads in progress. For relating the number of file downloads to the buffer content distribution, the TCP bandwidth-delay product is applied as follows:

$$w \geq C \cdot RTT, \tag{19}$$

where C represents the available bandwidth of a TCP connection (in bps), RTT the round-trip time (in seconds), w the TCP maximum window size (in bits). For considering a TCP connection with a continuous flow of data without waiting times, w should meet the above requirement [27]. This requirement can be easily met under the conditions considered in this paper: based on the TCP configurations used and an overly optimistic C of 5 Mbps, the RTT may exceed 10 ms, which supports the applicability of (19) in this context. Let the random variable X denote the number of TCP data segments in the WLAN AP transmission buffer. Then the number of TCP segments in the buffer can be approximated by the following relation:

$$X \approx N \cdot \lfloor w / X_{MSS} \rfloor, \tag{20}$$

where N is the number of file downloads in progress and X_{MSS} the TCP MSS. Then, in the spirit of (18), the probability distribution of X is approximated by: for $n = 0, 1, \ldots,$

$$P\left(X = n \cdot \lfloor w/X_{MSS} \rfloor\right) \approx \left(1 - \rho_{eff}\right) \rho_{eff}^{n}, \tag{21}$$

using the assumption that the TCP connections carrying the file download traffic have their maximum window size, w, of data segments in flight. Most of these data segments reside in the AP transmission buffer; one may be in transfer between the two stations and perhaps another segment may already have been received and should be acknowledged for when receiving the next (when using TCP delayed acknowledgements). Note that the dynamics of the TCP segments in the AP transmission buffer and the number of downloads in the network operate on different time scales.

3 Model validation for FTP downloads in WLAN

As WLAN networks all rely on the same MAC protocol basis, the MAC parameters of our model are based on the IEEE 802.11b standard amendment for its wide availability and lower computational requirements involved in high-load simulation validation. In contrast to previous contributions [30, 29, 25] we validate our model with OPNET Modeler (v14.5) [19], rather than using ns-2 simulator. OPNET contains a standard library of detailed WLAN models, including an AP, that may also serve as application server, and wireless stations that are used for downloading files from the AP. Our simulated network consists of 11 nodes; an AP is surrounded by 10 stations spaced at an equal distance of 15 meters. All nodes are configured to use the 802.11b 11 Mbps transmission rate. FTP requests arriving from the stations at the AP are handled by the application-layer of the AP. As these simulation models take many more parameters into consideration than our proposed analytic model, the simulation scenarios require careful configuration. To this end, the impact of the central processing unit (CPU) on file transfers is eliminated by assuming infinitely fast CPUs in all devices. Our model is validated for the widely used Reno TCP configuration [32] and an enhanced version of Reno indicated as *Full-Featured*. Our Full-Featured TCP configuration uses Selective Acknowledgements (SACK) [24] and has a slightly smaller MSS (due to the use of timestamps) to fit in the 1500 bytes that are used as the wireless LAN service data unit. Another important aspect is the buffer space on the network interface of the AP which contains all packets to be transmitted; a vast amount of packets may queue-up and overflows have a major impact on the results. In our simulations the AP buffer was set sufficiently large to avoid packet drops.

We have conducted extensive experimentation to validate our model. Table 2 summarizes the simulation settings specific to our experiments. Simulation runs were performed for two different TCP configurations, up to five file-size distributions, for two mean file size settings and up to seven load values. Each run was executed until a sufficiently small 95% Confidence Interval (CI) was obtained, often requiring durations up to 1000 hours of real time simulation for higher load values. Each of the outcomes is based on averages in the order of 2.6×10^6 samples (excluding an extensive warm-up period of up to 1.4×10^5 observations). In total, over 190 runs were performed for our validation, some of which have consumed up to 299 hours of computing time on state of the art simulation servers. The results for several representative examples are outlined below.

Table 2: Simulation settings

Variable	Setting
X_{FTPget}	4096 bits
$X_{FTPclose}$	64 bits
TCP_{stack}	$\{Reno, Full-Featured\}$
X_{MSS}	$\{11680(Reno), 11584\}$bits
$X_{tcp/ip}$	$\{320(Reno), 416\}$ bits
w	70080 bits (8760 bytes)
X_{file}	$\{200kBytes, 1MBytes\}$
$\frac{1}{\lambda_{200kb}}$	$\{0.48, ..., 0.38\}$sec, step 0.02 sec.
$\frac{1}{\lambda_{1Mb}}$	$\{2.4, 2.3, 2.2, 2.1.2.0, 1.9, 1.85\}$ sec.

3.1 Comparison

To illustrate the accuracy of our model we have, first, reproduced the models from Roijers et al. [29] and from Sakurai and Hanley [30], to match with the original outcomes specified in their contributions. Subsequently, we have parameterized the WLAN MAC models in accordance to one of our simulation settings (using long PHY preamble, TCP delayed acknowledgements and a TCP MSS of 1448 bytes). Finally, the obtained MAC-layer throughput is used in a $M/G/1$ PS model, but in contrast we use Poisson arrivals instead of applying admission control on the number of flows in the network [29], or admitting stations to have only one TCP flow [30]. In Figure 2 the expected file download time as a function of the effective load, ρ_{eff}, is shown for various models: (1) the analytic model from [29] (indicated as *RBF*), (2) the analytic model from [30] (*SH*), (3) our analytic model presented in Section 2.2 (*Our Model*), (4) a simplified version of our model (*Basic Model*) that ignores all detailed overhead due to the additional FTP, TCP interactions and transmitting the last cycle, and (5) simulations using exponential file-size distributions.

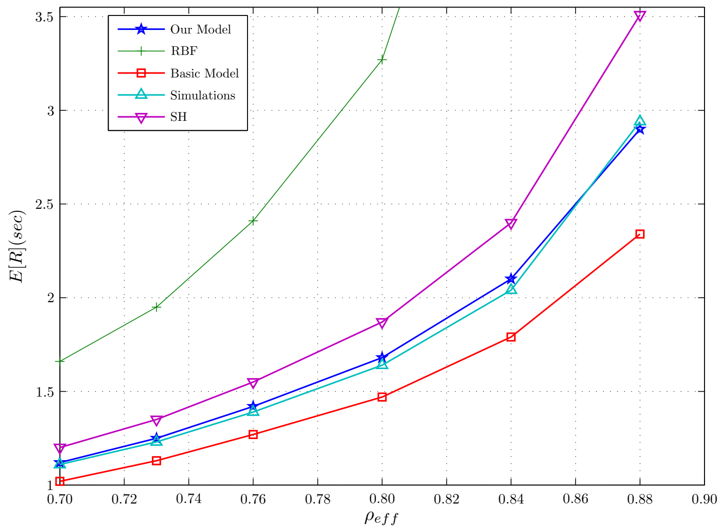
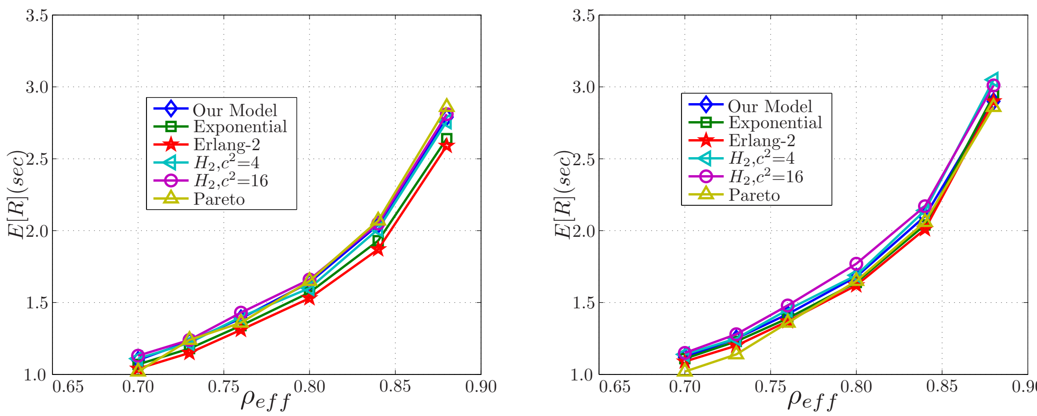


Figure 2: Average file download response time, $E[R]$, as a function of the effective load, ρ_{eff}, for various analytic models together with the simulation outcomes for the exponential distribution (using mean file size of 200 kBytes and Full-Featured TCP configuration).

The results in Figure 2 lead to a number of interesting observations. First, our analytic model matches very closely with the simulation results with an error of $1 - 3\%$. Second, the results strongly outperform those from [29] (indicated as RBF) in which the attained file download times are severely over-estimated. The outcomes from [30] (indicated as SH) provide a closer match compared to those from [29], with an error of approximately 8% for an effective load of 0.70. For this load value the errors are far less sensitive to the obtained WLAN MAC throughput from the model than for 0.88, where relatively small differences in obtained WLAN throughput will cause large errors of approximately 20%. It should, however, be noted that the modelling assumption in [30] of having only one TCP flow per station is not respected in our validation and that our resulting effective load of approximately 0.9 is much larger than the 0.26 and 0.31 considered by [30] that is using traffic sources based on idle periods rather than Poisson arrivals. A third observation made from Figure 2 is that the model outcomes are inaccurate when the overhead associated to TCP and FTP session set-up and closure is ignored (see also Remark 3 in Section 3.3). Indeed, for an effective load of 0.70 fair outcomes are obtained with an error of 8%. However, when the effective load is increased to 0.88 the error becomes larger than 20%. These observations confirm our expectation, formulated in Section 2, that our model leads to more accurate outcomes for the circumstances we consider in this paper.

3.2 Near insensitivity

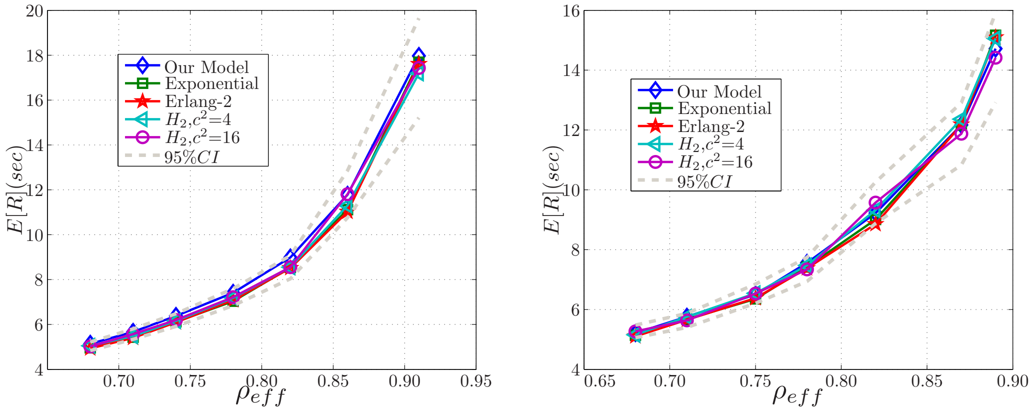
Our analytic model, defined in Section 2, adopts the insensitivity property of the mean response times with respect to the file-size distributions from the $M/G/1$ PS model. This raises the question to what extent the real download times are indeed insensitive. Figure 3 and 4 show the average file download response time, $E[R]$, as a function of the effective load, ρ_{eff}, for a range of parameter combinations, including light- and heavy-tailed file-size distributions for different mean values and light- and heavy-load scenarios. We observe that the analytic results closely match those from the simulation for a wide range of model parameters. The results also suggest that there is sensitivity with respect to the file-size distribution, but this sensitivity is quite weak (with errors typically of $1-3\%$ and no more than 8%). This observation is also in line with Roberts [28]. However, it should be noted that perfect fairness will not be obtained. TCP's slow-start mechanism causes bias against small file sizes; during the slow-start phase TCP connections do not attain a fair share of the medium capacity. File-size distributions with a higher variability will also perceive greater influence because small files are more predominant (on which the TCP slow-start delay has relatively more impact). Despite this biased behaviour of TCP, the influence on the simulation results is rather limited.



(a) Using TCP Reno implementation.

(b) Using Full-Featured TCP configuration.

Figure 3: Average file download response time, $E[R]$, as a function of the effective load, ρ_{eff}, using exponential, Erlang-2, Pareto (with shape parameter = 1.33) and hyper-exponential file-size distributions with a mean value of 200 kBytes. The obtained 95% Confidence Intervals (CI) are at most 1.9% and therefore omitted.



(a) Using TCP Reno implementation. (b) Using Full-Featured TCP configuration.

Figure 4: Average file download response time, $E[R]$, as a function of the effective load, ρ_{eff}, using exponential, Erlang-2 and hyper-exponential file-size distributions with a mean value of 1 MByte. The 95% CI values are shown for the hyper-exponential (H_2, with $c^2 = 16$) file-size distribution which provides the largest ones observed.

To assess the impact of the file-size distribution on the AP buffer content distribution, Figure 5 shows the CCDF (on a log-scale), obtained both from simulation, and from the analytic model using (21) under the assumption that condition (19) is satisfied (as is the case for the simulation settings outlined in Table 2). The results suggest that there is no significant dependence between the AP buffer content distribution and the file-size distribution. Moreover, the results show that our analytic model accurately predicts the simulation outcomes. See Remark 2 in Section 3.3 for some additional comments on the AP buffer size.

3.3 Engineering guidelines for practical application

Any approximation method, by definition, has parameter combinations where the results become less accurate. This raises the need for simple practical guidelines to determine for which combinations of parameters the model is valid. To this end, we have investigated the combined impact of the effective load, the TCP maximum window size, and the WLAN AP buffer size. The results are outlined below. We re-emphasize that the dynamics of the protocol stack, ranging from the application to the physical layer, and its interactions are extremely complex, while practice calls for simple guidelines, and therefore should be judged from that perspective.

For sufficiently large buffers, the TCP retransmissions are solely due to time-

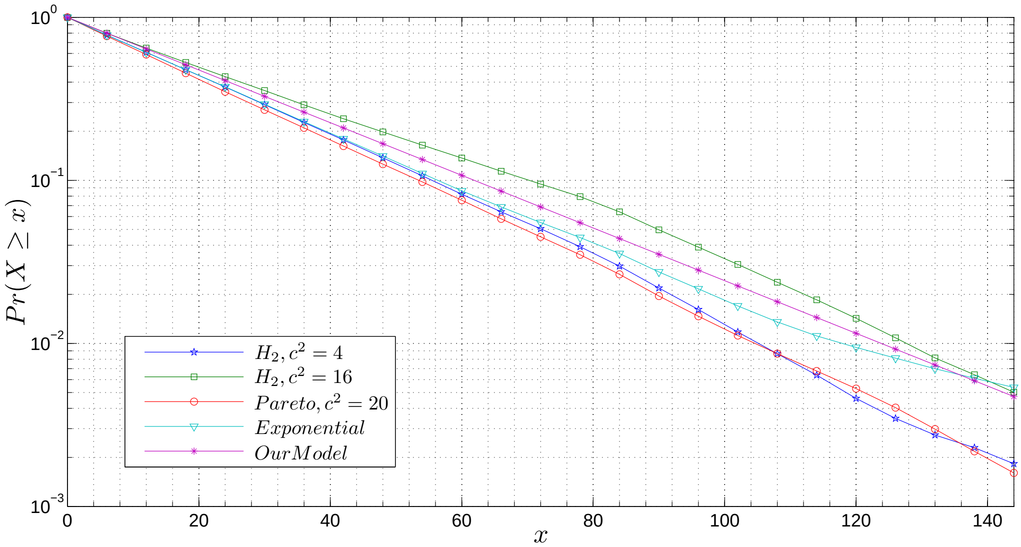


Figure 5: CCDF of the WLAN AP buffer content (x, in packets) from model and simulations, using exponential, Pareto (with $c^2 = 20$) and hyper-exponential file-size distributions for the same mean file size and TCP configuration as in Figure 3b with $\rho_{eff} = 0.8$.

outs and not to buffer overflow. Extensive experiments reveal that good results are obtained as long as:

$$\rho_{eff} \leq 0.9. \tag{22}$$

When ρ_{eff} exceeds 0.9, TCP retransmissions start to have a noticeable influence, as the segment delay more than rarely exceeds the TCP timeout value due to severe queuing at the AP buffer (see also Figure 6 below). For smaller values of the AP transmission buffer size Q (in segments of size X_{MSS}), it is of interest to know which *combinations* of Q and ρ_{eff} the model is applicable. Assuming that condition (19) is fulfilled, we approximate the packet loss probability, P, as:

$$P \approx Pr\left(X \geq Q\right) = \rho_{eff}^{Q}, \tag{23}$$

where X is related to the number of downloads in progress N by approximation (20) and the equality follows directly from (21). This immediately leads to the following rule of thumb for applicability of the model:

$$\rho_{eff}^{Q} \leq \alpha. \tag{24}$$

Extensive simulations suggest that accurate outcomes are obtained for α in the range of 1%-3%. For larger values of α TCP retransmissions have a profound impact on the download response time due to buffer overflows.

To illustrate the behaviour that is observed when the effective load is extremely high (and (22) is not respected), additional simulations have been conducted. Figure 6 shows the outcomes of our analytic model and the simulations for the case of exponential file-size distributions. We observe that when the effective load ρ_{eff} exceeds (say) 95% the results tend to become inaccurate. Nonetheless, note that in practice sustained extreme load values (in excess of 90%) hardly occur in well-dimensioned systems. This is in line with the observations of Ben Fredj et al. [15], stating that networks with such extreme load values are severely under-dimensioned and call for other approaches.

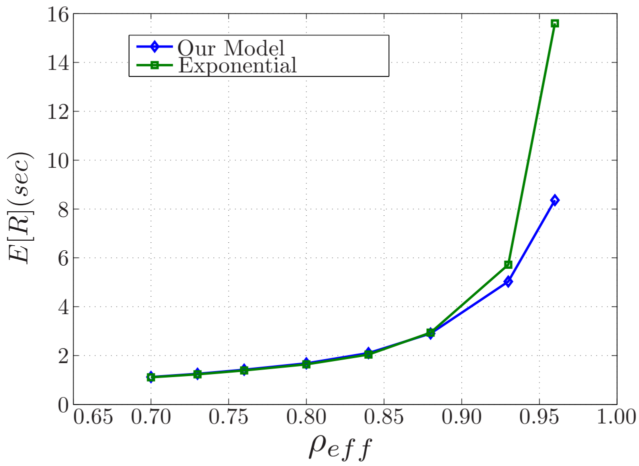
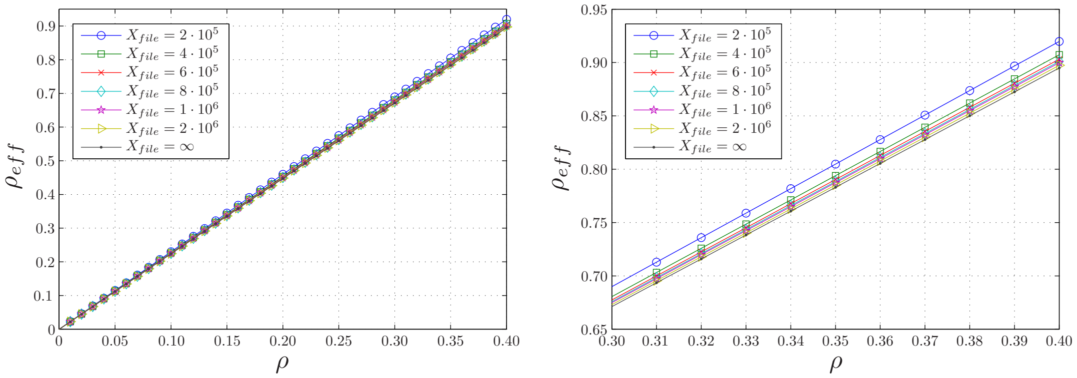


Figure 6: Average file download response time, $E[R]$ from the analytic model and from simulations, as a function of the effective load, ρ_{eff} including overload values. The outcomes are obtained under the same conditions as for Figure 3b.

We end this section with a number of remarks.

Remark 1 (Effective load versus offered load): The analytic model derived in Section 2.2 yields a linear relationship between the load, ρ, and the effective load, ρ_{eff}, on the network. Here, the load is defined by $\rho := \lambda\beta$ with λ the flow arrival rate and where β is the mean time to transmit files with mean size X_{file} on the WLAN medium at its transmission rate R. Note that R equals R_b for IEEE 802.11b-based systems and $R_{a/g/n}$ for IEEE 802.11a/g/n-based systems, respectively.



(a) From low to high effective load. (b) From medium to high effective load.

Figure 7: Relationship in analytic model between load, ρ, and effective load, ρ_{eff}, for different values of the mean file size, X_{file} (in bits), under the same conditions as for Figure 3a.

Figure 7 shows the linear relation between the load of the effective load for various values of X_{file} that all meet in the origin. When X_{file} grows large, the slope converges to $T_{cycle}R/2X_{MSS}$, with R the transmission rate of the medium. This result is a direct consequence of (12). An intuitive explanation is that the influence of the TCP setup, closure and FTP commands on the response times tends to become negligible as the file size becomes large.

Remark 2 (AP transmission buffer): For determining the AP buffer requirement using (24), condition (19) should be met. Otherwise an upper bound for the buffer size is obtained because the number of segments in flight will be lower than the number that fit in the maximum TCP window. It is important to note that the number of flows in the network is not limited, but geometrically distributed. This means that even for very large AP transmission buffers, overflows may occur. Furthermore, if condition (24) cannot be satisfied for the indicated range of α, the packet loss probability due to AP buffer overflow becomes substantial, model refinements taking TCP retransmissions explicitly into account are required, which opens up a challenging topic for further research.

Remark 3 (Applicability of the model to small file sizes): Our PS-based analytic model presented in Section 2 implicitly assumes that the files are transmitted in a sufficiently large number of TCP segments to justify the capacity sharing effect. In this context, Roberts [28] has indicated that the PS model does not capture all details of flows in networks that are controlled by TCP, where the slow-start algorithm causes a bias against small values of the file size and the additive increase multiplicative decrease introduces a bias against flows with high round-trip times. This limits the applicability of our model to file sizes that are representative for FTP traffic and requires several tens of TCP segments. For file sizes consisting of a few TCP segments only, the performance is dominated by the effects of round-trip times and TCP slow-start, which limits the applicability of the model for small file sizes.

Remark 4 (Relation to Bianchi-type of model): Initially Bianchi's model [8], that uses a fixed-point approach, served as a basis for flow-level performance models on WLAN networks [22, 25]. As opposed to using (RTS/CTS) channel reservations, the aggregated WLAN throughput in basic access mode, when used without TCP, strongly depends on the number of active stations [8, 22]. Extensive validation shows that this is not the case when the data flows use TCP (as considered in the present paper) and are transferred over a WLAN that operates in basic access mode; the absence of admission control (either per station or for the whole network) has no significant impact on the performance of file transfers for a range of different parameter settings (as long as the requirements for the AP buffer and maximum effective load from (22) and (24) are respected). In fact, the simulation outcomes confirm that few stations contend for the medium at the WLAN MAC level. We observe that (1) the number of collisions on the medium is in line with what has been incorporated in the model (in Equation (6) for instance), (2) the mean backoff interval drawn by all WLAN STAs is close to half of the *minimum* backoff window size (Cw_{min}), and (3) these two parameters do not depend significantly on the load (up to effective load values of approximately 0.9) nor the number of stations in the network. This implies that the medium contention is low, which reduces the need for including the fixed-point approach from [8] (in which all stations are assumed to always have a packet to transmit) for accurately modelling file transfers. Furthermore, large improvements in model accuracy are obtained when increasing the level of detail and applying correct parameterization to closely match outcomes of a commercially available simulation environment [19].

Remark 5 (TCP-stack implementations): It should be noted that TCP-stack implementations vary from one operating system to another and may sometimes even allow user modifications for parameter tuning. In fact, recent operating systems, such as Windows Vista [13], have appeared that apply automatic parameter tuning schemas, e.g. for setting the maximum receive window size to better utilize the available capacity on networks with a large bandwidth-delay product. These changes to the TCP-stack and the general trend towards creating larger receive window sizes (as proposed by [20]) may impact the performance and sensitivity of file transfers [25]. However, limited support of the window scaling techniques from [20] as well as the penetration rate of Vista may still lower the influence of these recent advances. Further work is needed to assess the impact of automatic parameter adaptation techniques in TCP.

Remark 6 (Backward compatibility): Practical WLAN deployments rely on the same IEEE 802.11 basic standard [1] for their fair method of medium access control, a property that is adopted by our proposed analytic model. Under those circumstances stations may operate on different transmission rates or even on a different standard to support higher data rates or prioritization techniques. In this context we refer to [26] for a detailed description on the use of the 802.11 a/b/e/g/n standards and a mixture thereof with the influence of backward compatibility.

4 Topics for further research

The mapping from packet-level to the flow-level performance model based on combining the $M/G/1$-PS model with load ρ_{eff} enables us to use known analytic results for PS models. It is an interesting topic for further research to generalize our approach to formulating an effective service time and resulting effective load for a wider range of access networks. Furthermore, a wealth of literature is available on PS models, we refer to [7, 33, 9, 11] and references therein for results on sojourn times in PS models. For example, rather than focusing on the unconditional expected response times, see (17) and the simulation results, we may extend the results to the conditional expected response times, the conditional and unconditional variance, and in some cases even for the tail probabilities of the response times, based on known results for the $M/G/1$ PS model [12]. To assess the usefulness of such results is a challenging topic for further research.

In the present paper, the focus is on the performance of data transfers in the download direction. However many applications generate large amounts of traffic in the upstream direction, which raises the question to what extent our analytic model is also applicable to networks where a mixture of up and download traffic is present. In the context of comparing the performance of FTP download and upload traffic, the authors in [30] have observed that the received throughput of upstream and downstream flows are "virtually identical" under circumstances that are similar to those considered in this paper. This suggests that our model may also accurately predict the performance of traffic in the upstream direction, which is a challenging topic for further research.

Another area of further research is the analysis of traffic splitting strategies that aim to optimize application performance by distributing the traffic of a single application over the multiple networks that may be available to user equipment and thus used concurrently. We refer to [18] and [16] for results on concurrent multi-path transfer approaches that divide larger traffic flows into smaller ones that are transported over multiple networks in parallel. To assess the performance benefits of these approaches using $M/G/1$ PS models may reveal new insights on their performance, in particular in the presence of (non-optimized) background flows in those networks. This is an area for further exploration [16].

5 Acklowledgments

The work reported in this paper was supported by the Netherlands Organisation for Scientific Research (NWO) under the Casimir project: Analysis of Distribution Strategies for Concurrent Access in Wireless Communication Networks. We thank the four anonymous referees for carefully reading our manuscript, their comments have greatly improved this paper.

References

[1] ANSI/IEEE Standard 802.11. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications. 1999.

[2] IEEE Standard 802.11a. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: High speed physical layer in the 5 GHz band. 1999.

[3] IEEE Standard 802.11b. Higher Speed Physical layer (PHY) extension in the 2.4 GHz band. September 1999.

[4] IEEE Standard 802.11g. Part 11: Wireless LAN Medium Access control (MAC) and Physical Layer (PHY) specifications: Further higher-speed physical layer extension in the 2.4 GHz band. June 2003.

[5] IEEE Standard 802.11n. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer specifications Enhancements for Higher Throughput. October 2009.

[6] J.V.L. Beckers, I. Hendrawan, R.E. Kooij, and R.D. van der Mei. Generalized processor sharing models for internet access lines. In *Proceedings of IFIP Conference on Performance Modelling and Evaluation of ATM and IP networks*, pages 101–112, Budapast, 2001.

[7] J.L. van den Berg. *Sojourn Times in Feedback and Processor Sharing Queues*. PhD thesis, 1990.

[8] G. Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, 18(3):535–547, 2000.

[9] S. Borst, R. Núñez-Queija, and B. Zwart. Sojourn time asymptotics in processor-sharing queues. *Queueing Systems: Theory and Applications*, 53(1-2):31–51, 2006.

[10] S.C. Borst, O.J. Boxma, and N. Hegde. Sojourn times in finite-capacity processor-sharing queues. In *Proceedings NGI 2005 Conference*, 2005.

[11] O.J. Boxma and H. Daduna. Sojourn times in queueing networks. In *Stochastic Analysis of Computer and Communication Systems, ed. H. Takagi (IFIP, North-Holland, Amsterdam, 1990)*, pages 401–450.

[12] E. G. Coffman, Jr., R. R. Muntz, and H. Trotter. Waiting time distributions for processor-sharing systems. *Journal of the ACM*, 17(1):123–130, 1970.

[13] Microsoft Corporation. Next Generation TCP/IP Protocols and Networking Components, June 2009. `http://technet.microsoft.com/en-us/library/cc754287.aspx`.

[14] N.T. Dao and R.A. Malaney. Throughput performance of saturated 802.11g networks. In *AUSWIRELESS '07: Proceedings of the 2nd International Conference on Wireless Broadband and Ultra Wideband Communications*, 2007.

[15] S. Ben Fredj, T. Bonald, A. Proutiere, G. Régnié, and J.W. Roberts. Statistical bandwidth sharing: a study of congestion at flow level. *SIGCOMM Computer Communication Review*, 31(4):111–122, 2001.

[16] Y. Hasegawa, I. Yamaguchi, T. Hama, H. Shimonishi, and T. Murase. Deployable multipath communication scheme with sufficient performance data distribution method. *Computer Communications*, 30(17):3285–3292, 2007.

[17] G.J. Hoekstra and R.D. van der Mei. On the Processor Sharing of File Transfers in Wireless LANs. In *Proceedings of the 69th IEEE Vehicular Technology Conference, VTC Spring 2009*, Barcelona, Spain, 26-29 April 2009.

[18] H.Y. Hsieh and R. Sivakumar. A transport layer approach for achieving aggregate bandwidths on multi-homed mobile hosts. *Wireless Networks*, 11(1):99–114, January 2005.

[19] OPNET Technologies Inc. Opnet modeler, May 2009. `http://www.opnet.com/solutions/network_rd/modeler.html`.

[20] V. Jacobson, R. Braden, and D. Borman. TCP extensions for high performance. RFC 1323, Internet Engineering Task Force, May 1992.

[21] F.P. Kelly. *Reversibility and Stochastic Networks*. John Wiley and Sons Ltd., Chichester, 1979.

[22] R. Litjens, F. Roijers, J.L. Van den Berg, R.J. Boucherie, and M.J. Fleuren. Performance analysis of wireless LANs: An integrated packet/flow level approach. In *Proceedings of the 18th International Teletraffic Congress - ITC18*, pages 931–940, Berlin, Germany, 2003.

[23] P. Mahasukhon, M. Hempel, S. Ci, and H. Sharif. Comparison of throughput performance for the IEEE 802.11a and 802.11g networks. In *Proceedings of the 21st International Conference on Advanced Information Networking and Applications (AINA 2007), May 21-23, 2007, Niagara Falls, Canada*, pages 792–799. IEEE Computer Society, 2007.

[24] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP Selective Acknowledgment Options. RFC 2018, Internet Engineering Task Force, October 1996.

[25] D. Miorandi, A.A. Kherani, and E. Altman. A queueing model for HTTP traffic over IEEE 802.11 WLANs. *Computer Networks*, 50(1):63–79, 2006.

[26] F.J.M. Panken and G.J. Hoekstra. Multi-service traffic profiles to realise and maintain QoS guarantees in wireless LANs. *Computer Communications*, 32(6):1022 – 1033, 2009.

[27] A. Riedl, T. Bauschert, M. Perske, and A. Probst. Investigation of the M/G/R Processor Sharing Model for Dimensioning of IP Access Networks with Elastic Traffic. In *Proceedings of the First Polish-German Teletraffic Symposium PGTS 2000*, Dresden, 2000.

[28] J.W. Roberts. A survey on statistical bandwidth sharing. *Computer Networks*, 45(3):319–332, 2004.

[29] F. Roijers, J.L. van den Berg, and X. Fang. Analytical modelling of TCP file transfer times over 802.11 wireless LANs. In *Proceedings of the 19th International Teletraffic Congress - ITC19*, Beijing, China, 2005.

[30] T. Sakurai and S. Hanley. Modelling TCP flows over an 802.11 wireless LAN. In *Proceedings of European Wireless Conference*, 2005.

[31] W.R. Stevens. *TCP/IP Illustrated, Volume 1: The Protocols*. Addison-Wesley Professional, New York, NY, USA, fourth edition, 1993.

[32] Y. Tian, K. Xu, and N. Ansari. TCP in wireless environments: problems and solutions. *IEEE Communications Magazine*, 43(3):27–32, 2005.

[33] J.L. van den Berg and O.J. Boxma. The M/G/1 queue with processor sharing and its relation to a feedback queue. *Queueing Syst. Theory Appl.*, 9(4):365–402, 1991.

[34] Y. Wu, C. Williamson, and J. Luo. On processor sharing and its applications to cellular data network provisioning. *Performance Evaluation*, 64(9-12):892–908, 2007.

.