Engineering Elastic Traffic in TCP-Based Networks: Processor Sharing and Effective Service Time

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

[1]Innovation Research & Technology, Thales Nederland B.V., Huizen, The Netherlands
[2]CWI, Department of Stochastics, Amsterdam, The Netherlands
[3]VU University Amsterdam, Department of Matlhematics, The Netherlands

***Abstract*—Today, a wide range of networks exist that provide users with data services using TCP. For performance engineering of such networks there is a need for simple but accurate models to predict the performance under anticipated load conditions. This is a challenging task, because the combined packet-level dynamics of the network-protocol stack are highly complicated. First, we explain how the packet-level dynamics and protocol overhead of the multiple communication layers can be successfully captured by a single parameter, called the *effective service time* (EST), and give a full parameterization that explicitly expresses the EST in terms of the network parameters. Second, we show how the transfer-time performance can be modeled by a classical M/G/1 Processor Sharing (PS) model, but where the service time is replaced by the EST. Finally, extensive test-lab experimentation shows 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, Flow-level Performance, Transfer Times, Transmission Control Protocol, Parameterization.**

I. Introduction

Processor Sharing (PS) models are frequently used to describe the bandwidth sharing of elastic traffic in TCP-based networks with notable contributions in this area from [1] for accounting TCP bandwidth and [2] on the impact of user behavior. A particularly attractive feature of PS models is that they abstract from the complex packet-level details of the network, but at the same time maintain the essential factors that determine the data transfer-time performance of elastic data flows. Moreover, the theory of PS models is well-matured and has been successfully applied to model the flow-level behavior of a variety of communication networks, including CDMA 1xEV-DO [3], WLAN [4], UMTS-HSDPA [5] and ADSL [6]. In this context we refer to two key contributions [7], [8] that consider user-level performance in wireless data channels. Despite the fact that PS models are often used to describe the transfer-time behavior of TCP-based networks, hardly any results are known about *how to parameterize* the model, i.e., how to translate the packet-level network parameters (e.g., at the MAC, network, transport and application layer) into the parameters of the corresponding PS model.

Motivated by this, the main goal of this paper is to make a first step towards filling this *gap* between PS models on the one hand and TCP-based networks on the other hand by providing a full and explicit parameterization of the model. Evidently, the transfer-time performance generally depends on the specific protocols, and their parameter choices, used at the different network layers. In this paper, we focus on the widely used protocol stack FTP/TCP/IP/WLAN to exemplify the modeling approach, but emphasize that the applicability of this approach goes way beyond the protocol stack considered here.

In [4] a new analytic flow-level model was presented that translates the complex and detailed dynamics of the FTP/TCP/IP-stack over a WLAN (without admission control) into an explicit expression for the EST of a file of a given size. Based on the notion of the EST we can describe the flow-level behavior of TCP-based file transfers as an $M/G/1$-PS model where service time of a job of given size is replaced by the EST. Extensive *simulation* results in [4] demonstrated that the mean response times can be accurately predicted over a wide range of parameter combinations.

In this paper we extend the results in [4] in three important directions. First, we propose a refined model with a full parameterization for the FTP/TCP/IP/WLAN protocol stack that improves the performance of [4] for high-load circumstances by explicitly taking into account the influence of management traffic and frame encapsulation; these aspects manifest themselves especially in high-load circumstances. Experimental results show indeed a significant improvement in accuracy of the refined model compared to the basic model (see Section IV below). Second, to test the accuracy of the model-based performance predictions, we have implemented the system in a lab-test environment in order to compare the transfer-time performance of the network with the analytical results obtained from the corresponding PS model, while the validation results in [4] were based on simulations only. Third, we extend the validation experiments to the complete transfer-time *distribution*, and the distribution of the number of flows in the system. The experimental results for a wide variety of parameter settings, including light- and heavy-tailed file-size distributions and light- and heavy-load scenarios, show an excellent match between the analytic results and the test-lab experiments.

The remainder of this paper is organized as follows. In Section II we briefly discuss the parameterization of the EST model from [4], followed by model refinements suited for real network deployments. In Section III we translate this model

to into analytic results for the transfer-time distributions using results from the theory of M/G/1-PS systems. In Section IV we validate the model by comparing testbed experiments on the mean and the distribution of the file downloading times against our proposed analytic model. Finally, Section V contains concluding remarks and challenges for further research.

II. Effective service time model

For completeness, in Section II-A we outline the basic effective service time model (described in detail [4]). In Section II-B, we present the refined model.

A. Basic model

We consider a network consisting of several WLAN stations and one access point (AP), in which the stations are downloading files from an FTP server that is located close (with negligable propagation delay) to the AP. Each station generates FTP download requests according to a 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 acknowledgments. The model accounts for all overhead associated with a file download; the file transfer itself, the FTP commands and TCP handshake for opening and closing sessions on a WLAN network operating in basic access mode, using the Distributed Coordination Function (DFS). For further details on the assumptions of the effective service time model, we refer the reader to [4]. One can express the time spent by a WLAN station on transmitting a TCP *data* segment (including the IEEE 802.11 MPDU, IP and TCP overhead) of x bits and its associated MAC acknowledgment by $T_d(x)$ and T_c respectively:

$$T_d(x) = PHY + \frac{MAC + X_{hloh} + x}{TS}, \tag{1}$$

$$T_c = PHY + \frac{ack}{TS_c}, \tag{2}$$

where TS represents the WLAN transmission rate (in bps) for data segments and TS_c for the WLAN acknowledgments. The overhead associated with the higher-layer protocols of the IEEE 802.11 MAC is represented by X_{hloh}. In the case of using TCP and IP as higher-layer protocols X_{hloh} is set to $X_{tcp/ip}$ bits. 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 acknowledgment 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 T_p seconds:

$$T_{da}(x) = DIFS + T_d(x) + T_p + SIFS + T_c + T_p. \tag{3}$$

Note that the expressions for $T_d(x)$ and T_c from respectively (1) and (2) depend on the IEEE 802.11 standard used (see [4]). The total expected time of a transmission cycle of two TCP data segments and one TCP ACK segment can be written as follows:

$$T_{cycle} = 2T_{da}(X_{MSS}) + T_{da}(0) \tag{4}$$

$$+\frac{Cw_{min}\,(7Cw_{min}+8)\,\tau}{6(Cw_{min}+1)} + \frac{T_{col}}{Cw_{min}+1},$$

where

$$T_{col} = T_d(X_{MSS}) + T_p + EIFS, \tag{5}$$

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)).

Parameter	802.11a	802.11b	802.11g	802.11n
Cw_{min}(slots)	15	31	15	15
MAC(bits)	246	224	246	246
τ	9μs	20μs	9μs	9μs
SIFS	16μs	10μs	10μs	16μs
DIFS	34μs	50μs	28μs	34μs
EIFS	90μs	364μs	342μs	90μs
PHY	20μs	192μs	26μs	20μs
ack(bits)	134	112	134	134
T_p	1μs	1μs	1μs	1μs
TS(bps)	$54 \cdot 10^6$	$11 \cdot 10^6$	$54 \cdot 10^6$	$54 \cdot 10^6$
TS_c, TS_m(bps)	$24 \cdot 10^6$	10^6	$24 \cdot 10^6$	$24 \cdot 10^6$
STT	4μs	NA	4μs	4μs

TABLE I: Key parameters in IEEE 802.11 protocols.

The total time spent on average in a TCP set-up is:

$$T_{tcp\_setup} = 3T_{da}(0) + \frac{Cw_{min}(2Cw_{min}+1)\tau}{6(Cw_{min}+1)} + \frac{(2T_{shortcol} + T_{col})}{Cw_{min}+1}, \tag{6}$$

where

$$T_{shortcol} = T_d(0) + T_p + EIFS. \tag{7}$$

The average time spent on using the medium for the FTP GET request equals:

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

The file transfer is concluded by the transmission of the last data segment, which is immediately followed by an FTP close command of size $X_{FTPclose}$. The expected size of the last data segment of the file approximately equals $X_{MSS}/2$, and hence:

$$T_{lastcycle} = T_{da}(X_{FTPclose}) + T_{da}\Big(\frac{X_{MSS}}{2}\Big) \tag{9}$$

$$+T_{da}(0) + \frac{T_{halfMSSCol}}{Cw_{min}+1} + \frac{Cw_{min}\,(7Cw_{min}+8)\,\tau}{6(Cw_{min}+1)}$$

$$+\frac{1}{2}\Big(T_{da}(0) + T_{shortCol} + \frac{Cw_{min}(2Cw_{min}+1)\tau}{6(Cw_{min}+1)}\Big),$$

where

$$T_{halfMSSCol} = T_d\Big(\frac{X_{MSS}}{2}\Big) + T_p + EIFS. \tag{10}$$

After sending the last TCP data segment, the AP will contend with the station that attempts to transmit its last TCP ACK

and later sending its TCP FIN. The expected time to close the TCP connection can then be expressed as:

$$T_{TCP\_close} = 4T_{da}(0) + \frac{2Cw_{min}(4Cw_{min}+5)\tau}{6(Cw_{min}+1)} + \frac{2T_{shortcol}}{Cw_{min}+1}. \quad (11)$$

The expected time consumed by the FTP commands and the TCP session opening/closing as defined by (6), (8) and (11), can be expressed as:

$$T_{tcpftp_{OH}} = T_{tcp\_setup} + T_{FTP\_get} + T_{TCP\_close}. \quad (12)$$

Now, we obtain the EST of the transfer time of a file of size X_{file} (in bits) as observed at the application-layer by combining (4), (9), and (12):

$$EST(X_{file}) = \frac{\left(X_{file} - \frac{X_{MSS}}{2}\right)T_{cycle}}{2X_{MSS}} + T_{lastcycle} + T_{tcpftp_{OH}}, \quad (13)$$

Note that in our modeling approach, we have assumed the use of TCP with delayed acknowledgments.

B. Refined model

In the previous section, the main aspects for modeling file transfers over a WLAN network have been accounted for. In practice, however, the WLAN network also carries management traffic and applies frame encapsulation that is not commonly accounted for due to the limited impact this is considered to have on the overall network performance [9]. Moreover, the FTP application is in reality more sophisticated than described in the previous section. Although the model described previously closely matches the behavior of real networks for most purposes and for the simulation studies conducted, further model enhancements are proposed in this section to improve the analytic model specifically for testbed experiments that expose the network to high traffic loads. Exactly in these circumstances certain aspects that may seem of minor importance turn out to have a noticeable performance impact. In the following subsections the model enhancements for use in practical deployments are presented.

Frame encapsulation overhead

The Logical Link Control (LLC) layer as defined by the IEEE 802.2 standard can be used by underlying protocol (sub)layers, therefore the use of LLC is optional; some implementations of the IEEE 802.11 standard do not use the LLC layer (such as the OPNET modeler implementation), whereas other implementations (i.e., from equipment manufacturers like Cisco and Linksys) do use the LLC layer. In the latter case, two encapsulation methods may be applied, one method described in RFC 1042 [10] and the other in the IEEE 802.1H standard [11], both are derivatives of the IEEE 802.2 Subnetwork Access Protocol (SNAP) and introduce eight bytes of additional header information to the WLAN MAC payload. In the context of the presented performance model, there are no implications of the LLC/SNAP encapsulation as a higher-layer protocol other than a reduced medium efficiency. Since IEEE 802.11-based WLANs are capable of transmitting MAC frames with payloads up to 2304 bytes per frame, IP packets may still have a size of 1500 bytes, in accordance with RFC 1191 that defines the Path Maximum Transmission Unit (MTU) discovery for IP networks. Therefore, the overhead due to the LLC/SNAP encapsulation adds to the other encapsulation overhead and can be accounted for by using $X_{hloh} = X_{tcp/ip} + 64$ in (1).

FTP modeling

The current File Transfer Protocol (FTP) is standardized since 1985 in RFC 959 [12]. The first file transfer mechanisms were already proposed in 1971 and implemented on hosts at MIT, followed by many RFCs and other implementations. As opposed to the implementation in the OPNET network simulation environment, practical deployments operate in accordance with the methods outlined in RFC 959 as is done in the widely used ProFTPD [13]. To obtain more accuracy in modeling experimental deployments, more detailed FTP interactions are added to the analytic model that are outlined below. As we assume that all users have logged in to the FTP server already, the message sequences needed to setup a file download involve (cf. [14]) the transmission of an FTP passive command (PASV) with size $X_{ftp-pasv}$ by the station, followed by a 227 (entering passive mode) response from the server (with a piggy-backed TCP ACK) of size $X_{ftp-227}$ by the FTP server (and hence the AP) to confirm the passive file download mode. This two-way handshake is modeled similar to the first two sequences of the TCP setup in (6), but with the appropriate TCP segment payload values for the PASV request and the 227 response.

$$T_{ftp\_setup} = T_{da}(X_{ftp-pasv}) + T_{da}(X_{ftp-227}) + \frac{Cw_{min}(2Cw_{min}+1)\tau}{6(Cw_{min}+1)} + \frac{(T_{shortcol}+T_{col})}{Cw_{min}+1}. \quad (14)$$

Here, the idle time of the medium due to backoff is expected to be the minimum of two backoff observations because the AP and the station have both a packet to transmit. The PASV request may collide with a TCP data segment from the AP with the probability of both stations drawing the same backoff interval, whereas the 227 response by the AP may only collide with the smaller packets from the stations. After receiving the 227 (entering passive mode) response from the FTP server, the station initiates a new TCP connection for the data transfer and issues an FTP retrieve command (RETR) with size $X_{ftp-retr}$ to obtain a certain file. As soon as the TCP connection is established, the station issues the FTP RETR command and will receive a 150 (opening binary mode data connection) response message of size $X_{ftp-150}$ from the FTP server. Since the station's backoff time does not inhibit the AP from using the medium, the backoff time is in this case not modeled explicitly. Hence, the average time spent on using the medium for the FTP RETR Request and Response (RR) equals:

$$T_{ftp\_rr} = T_{da}(X_{ftp-retr}) + T_{da}(X_{ftp-150}) + \frac{Cw_{min}(2Cw_{min}+1)\tau}{6(Cw_{min}+1)} + \frac{T_{col}+3T_{shortcol}}{2(Cw_{min}+1)}. \quad (15)$$

Directly after the 150 response (opening binary mode data connection) is transmitted by the FTP server, the TCP transmission cycle is repeated up to the point where the last cycle is transmitted. Unlike the model in Section II-A we assume in the present model that the user's FTP session remains active and

does not require the transmission of an FTP closure command. Hence the last transmission cycle may consist of one or two TCP data segments, with equal probability, and the size of the last TCP data segment will on average be $X_{MSS}/2$ bits. We therefore conclude that the last TCP transmission cycle transports on average X_{MSS} bits in an expected time equal to:

$$\tilde{T}_{lastcycle} = \frac{T_{da}\left(X_{MSS}\right)}{2} + T_{da}\left(\frac{X_{MSS}}{2}\right) \tag{16}$$
$$+T_{da}(0) + \frac{T_{halfMSSCol}}{Cw_{min}+1} + \frac{Cw_{min}\left(11Cw_{min}+13\right)\tau}{12(Cw_{min}+1)}.$$

In (16) the last term of $\tilde{T}_{lastcycle}$ represents the idle time of the medium due to backoff and accounts for the fact that the last cycle may consist of one or two TCP segments. As a result, two or three backoff periods may occur during the last cycle with equal probability. After the station has acknowledged the last TCP data segment, it will close the TCP data connection by transmitting a TCP FIN that is acknowledged by the FTP server, followed by an FTP 226 (transfer complete) response of size $X_{ftp-226}$ to indicate that the transfer has completed. Finally, the station acknowledges this response, which concludes the data transfer. Similar to the TCP connection setup (6) and the FTP retrieve request (15) the AP contends with at least one station for the medium, causing the medium idle time to be equal to the minimum of two backoff windows and possible collisions occur with small packets. Accordingly, the station's backoff does not inhibit the medium from being used and the transmissions may collide with the larger TCP data segments from the AP. The expected time to close the FTP data transfer can be approximated by:

$$\tilde{T}_{TCP\_close} = 3T_{da}(0) + T_{da}(X_{ftp-226}) \tag{17}$$
$$+\frac{2Cw_{min}(2Cw_{min}+1)\tau}{6(Cw_{min}+1)} + \frac{2T_{shortcol}+2T_{col}}{Cw_{min}+1}.$$

The time consumed by transmitting the FTP overhead and TCP session opening/closing as defined by (6), (14), (15) and (17) is calculated by:

$$\tilde{T}_{tcpftp_{OH}} = T_{ftp\_setup} + T_{tcp\_setup} + T_{ftp\_rr} \tag{18}$$
$$+ \tilde{T}_{TCP\_close}.$$

Now, we obtain the EST of the transfer of a file of size X_{file} (in bits) as observed at the application-layer for the enhanced analytic model by combining (4), (16), and (18):

$$E\tilde{S}T(X_{file}) \tag{19}$$
$$= \frac{\left(X_{file} - \frac{X_{MSS}}{2}\right)T_{cycle}}{2X_{MSS}} + \tilde{T}_{lastcycle} + \tilde{T}_{tcpftp_{OH}}.$$

Recall that in our modeling approach, we have assumed the use of TCP with delayed acknowledgments.

WLAN beacon frames

Many performance models have appeared in the literature that are validated by simulations (cf. [15], [16], [17], [18]) and have concentrated on the main aspects of the medium access control protocol. In [9] the authors observed that other performance models ignored the impact of among others the Timing Synchronization Function (TSF). In an infrastructure network, the APs are responsible for performing the power management function by distributing the time in certain management frames, called beacons. When APs are about to transmit a beacon frame, their local time is inserted into the beacon. The IEEE 802.11 standard mandates that the AP periodically transmits the beacons and that the receiving stations always accept the timing information received. Another purpose of the beacon frame is to announce the presence of an IEEE 802.11 network and its capabilities at regular intervals, called Beacon Intervals (BI). These time intervals are expressed in Time Units (TU) of 1.024 microseconds. Typically, beacon intervals are set to 100 TUs. The beacon frame is based on an IEEE 802.11 management frame, and requires for the transmission of a MAC MPDU of x bits at rate TS_m an amount of time equal to:

$$T_m(x) = PHY + \frac{x}{TS_m}. \tag{20}$$

Similar to the transmission of data frames, the AP follows the procedure described in the standard, [19], for the transmission of beacon frames. However, the receiving stations do not acknowledge the reception of the beacons that are broadcast by the AP. To this end, the transmission cycle of a beacon frame, T_{bc}, is approximated by using (20) as follows:

$$\tilde{T}_{bc} = DIFS + T_m(X_{beacon}) + T_p + \frac{Cw_{min}\tau}{2}, \tag{21}$$

where X_{beacon} is the size of the entire management frame including the beacon specific fields, as defined at page 45 of [19]. Although the transmission of beacons may be delayed due to CSMA deferrals, subsequent beacons have to be scheduled by the AP at the nominal interval, even on a busy network. Consequently, the effective service time of the file transfer as observed at the application-layer is stretched. The EST in the absence of beacon frames denoted $EST'(X_{file})$ that may be obtained from either (13) or (19) is treated as follows to obtain the $EST''(X_{file})$ that accounts for the presence of beacon frames:

$$EST''(X_{file}) = EST'(X_{file})\Big(1 + \frac{T_{bc}}{BI}\Big). \tag{22}$$

III. PS MODELS AND EFFECTIVE LOAD

To model the flow-level behavior 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 time units). In this model, incoming jobs immediately enter the system, thereby receiving a fair share of the available capacity. The load of the system is $\rho := \lambda\beta < 1$. For this classical model, a number of analytic results are known. For later reference, we formulate a number of analytical results. Let M denote the number of jobs in the system, and S denote sojourn time of an arbitrary job. Then the steady-state distribution of M is, for $m = 0, 1, \ldots$,

$$\Pr\{M = m\} = (1-\rho)\,\rho^m. \tag{23}$$

Moreover, using Little's formula we obtain the *unconditional* mean sojourn times

$$\mathbb{E}[S] = \frac{\mathbb{E}[M]}{\lambda} = \frac{\beta}{1-\rho}. \tag{24}$$

Furthermore, it is known that the *conditional* mean sojourn time of a job of size τ equals:

$$\mathbb{E}[S|\tau] = \frac{\tau}{1-\rho}. \tag{25}$$

Note that (23)-(25) are *insensitive* to the service-time distribution, in the sense that they depend on the service-time distribution through its mean β. Furthermore, for the case of exponential service-time distributions (normalized so that $\beta = 1$, without loss of generality), the sojourn-time distribution is (cf. [20]): For $t > 0$,

$$\begin{aligned} Pr\{S > t\} = 2 \int_0^{\pi} \{exp\{(-x[2\sqrt{\rho} - (1+\rho)cos(x)] \\ /[(1-\rho)sin(x)] - ((1-\rho)^2 t/(1+\rho-2\sqrt{\rho}cos(x))\}\}) \\ \{(1-\rho) \\ (1+exp\{-\pi[2\sqrt{\rho} - (1+\rho)cos(x)]/(1-\rho)sin(x)\})\}^{-1} \\ sin(x)dx. \end{aligned} \tag{26}$$

To translate the flow-level performance for WLAN file downloads (discussed above) into an M/G/1-PS model, we define the *unconditional* mean EST of an arbitrary flow (for the refined model) by

$$\beta_{eff} := \int_0^{\infty} EST''(\tau) dX_{file}(\tau), \tag{27}$$

where $EST''(\tau)$ is defined in (22). Moreover, define the *effective load* as follows:

$$\rho_{eff} := \lambda \cdot \beta_{eff}. \tag{28}$$

Similarly, the effective load is obtained for TCP implementations that use non-delayed acknowledgments or are based on a model enhancement suited for experimental purposes by using the EST definitions from (19) and (22). 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 (unconditional) file transfer time is modeled as the expected (unconditional) sojourn time in an M/G/1-PS model with mean service time β_{eff} and load ρ_{eff}:

$$\mathbb{E}[R] = \frac{\beta_{eff}}{1-\rho_{eff}}. \tag{29}$$

Thus, to apply the analytic model, the average file unconditional file-transfer time $\mathbb{E}[R]$ is obtained from (29), where β_{eff} and ρ_{eff} are given by (27) and (28).

Similarly, the distribution of the number of flows in the WLAN-system can be obtained using (23). If N denotes the number of file transfers in progress, then for $n = 0, 1, \ldots$,

$$\Pr\{N = n\} = (1-\rho_{eff})\,\rho_{eff}^n, \tag{30}$$

where ρ_{eff} is defined in (28).

IV. Model validation by testbed experiments

To validate whether the (parameterized) M/G/1-PS model accurately predicts the transfer-time performance in TCP-based networks, we have performed extensive test-lab experimentation. The results are outlined below.

A. Experimental setup

The experimental setup consists of two connected Ubuntu Linux-based PCs: one functioning as FTP server, and the other as FTP clients. These PCs are interconnected by two independent and similar access networks to measure the download response times of file transfers in each network separately and compare the obtained values with the expected file-download time from the analytic model. In the experimental setup, the PC with the FTP clients generates all download requests, according to independent Poisson processes. It is important to state that with a larger number of WLAN client devices there is, in addition to the AP, typically only one station contending for the medium at the same time, as reported in [16] and observed during the experiments in [21]. This justifies the choice of using one client device for our downloads rather than a large population. At the PC operating as FTP server, files are generated according to the following file-size distributions, with different values of the squared coefficient of variation of the job sizes (denoted c_B^2): deterministic (1 Mbytes, $c_B^2 = 0$), exponential ($c_B^2 = 1$), two-phase hyper-exponential with $c_B^2 = 4$ with (balanced means), and Pareto-2 (with $Pr\,(B > x) = \frac{1}{4x^2}$ for $x > 1/2$ and hence $c_B^2 = \infty$). All non-deterministic distributions consisted of 40.000 different files with mean size 200 kbytes that are retrieved in a random order by the FTP clients. In our testbed environment, the wireless access network consists of a Linksys (WAP54G) AP and a router (TP-Link 1043ND) that is transformed into an Ethernet bridge. Both devices are connected by a Mini-Ciruits power splitter (ZB8PD-4-S) to obtain a reproducible radio frequency environment with low interference and small propagation delay, T_p. The AP and the ethernet bridge use a modified firmware program, called OpenWrt, that is specifically designed for embedded devices such as residential gateways and routers. This firmware offers detailed WLAN-MAC configuration options. Table II summarizes the parameters that are specific for our testbed configuration.

Variable	Setting
$X_{ftp-pasv}$	48 bits
$X_{ftp-227}$	392 bits
$X_{ftp-retr}$	272 bits
$X_{ftp-150}$	704 bits
$X_{ftp-226}$	184 bits
X_{beacon}	584 bits
X_{MSS}	11680 bits
$X_{tcp/ip}$	320 bits
w	70080 bits (8760 bytes)
X_{file}	$\{2 \times 10^5, 1 \times 10^6\}$ bytes
$\frac{1}{\lambda_{200kb}}$	$\{0.48, 0.46, 0.44, 0.42, 0.40, 0.38, 0.36, 0.35\}$ seconds
$\frac{1}{\lambda_{1Mb}}$	$\{2.34, 2.22, 2.12, 2.02, 1.93, 1.85, 1.77, 1.70\}$ seconds
T_c	$11 \cdot 10^6$ bps
BI	100 Time Units (TUs)
T_p	10^{-9} seconds

TABLE II: Testbed environment and model parameters.

In Table II it is shown that the TCP stack in our testbed environment is configured to use an MSS, indicated as X_{MSS}, of 1460 bytes and a window-size, w, of 8760 bytes. The standard TCP implementation from our Ubuntu Linux is used, that is an implementation of the TCP protocol defined in RFC-793, RFC-1122 and RFC-2001 with the NewReno and SACK extensions. The TCP stack is configured accordingly to support the 8760-byte window-size and an MSS of 1460 bytes. Therefore we

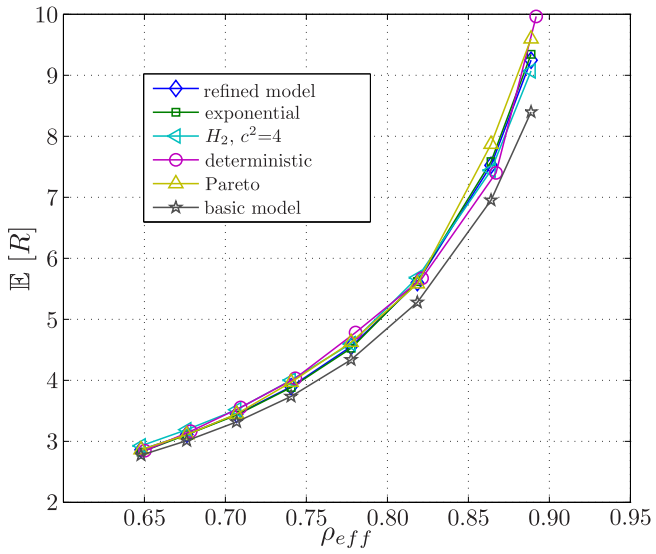

Fig. 1: Normalized mean transfer time $\mathbb{E}[R]$ as a function of ρ_{eff}, for different file-size distributions.

use in the analytic model 40 bytes of TCP/IP overhead of the higher-layer protocols, represented by variable $X_{tcp/ip}$. With regard to the IEEE 802.11 MAC parameters, slightly different settings are used in our experiments; (1) the AP is configured to broadcast beacon frames at a transmission rate, TS_m, equal to 10^6 bps (specified in Table I) with a commonly-used interval of 100 time units of 1.024 microseconds each, and (2) control frames (WLAN acknowledgments) are transmitted at a rate, T_c, equal to the transmission of data frames, $11 \cdot 10^6$ bps. In addition, the FTP commands and responses introduced in Section II-B are also specified in Table II.

B. Experimental results

Next we give an outline of the validation results.

B.1 Expected transfer-time and near-insensivity

Let us first look at the mean transfer time $\mathbb{E}[R]$, given by (29). Recall from Section III that in the M/G/1-PS model the mean sojourn time is insensitive to the job-size distribution, which suggests that in the real network the mean transfer time is at least near-insensitive to the file-size distribution. The check the accuracy of the model and the (in)sensitivity of the transfer-time performance with respect to the distribution of the file size, test-lab runs were performed for four different file-size distributions and eight different load values. Runs have been executed until a sufficiently small 95%-CI was obtained (with width less than 2.8% of the mean), requiring durations of over 140 hours to gather up to 1.450.000 observations per run. Figure 1 shows the experimental values of $\mathbb{E}[R]$ as a function of the effective load ρ_{eff} for the different job-size distributions, and the analytic results based on the basic model described in Section II.A and on the refined model discussed in Section II.B.

The results in Figure 1 lead to a number of interesting observations. First, they demonstrate that the analytic results closely match the outcomes from the experimental testbed for a broad range of model parameters. Second, we observe that there is no significant dependence of the mean transfer time in our experiments on the file-size distribution, which confirms the (near-)insensitivity result (29) suggested by the M/G/1-PS model. Third, the results show that the refined model indeed outperforms the basic model, particularly under heavy load scenarios, as it should. We re-emphasize the importance of these observations, which extend the results in [4] to real network equipment and underlines the usefulness of PS models to predict the transfer-time performance of elastic traffic streams.

B.2 Probability distribution of the file-transfer times

For the special case of exponential job sizes, an exact expression for the tail probability of the M/G/1-PS model is given in (III). This result suggests that we can approximate the $Pr\{R > t\}$ in the real network by using (III), where ρ is replaced by ρ_{eff}, defined in (28). To validate this, Figure 2 shows the tail probabilities, both by using (III) and by lab experiments, for a system with exponential file-size distributions with mean $\beta_{eff} = 0.312$ and effective load $\rho_{eff} = 0.78$.

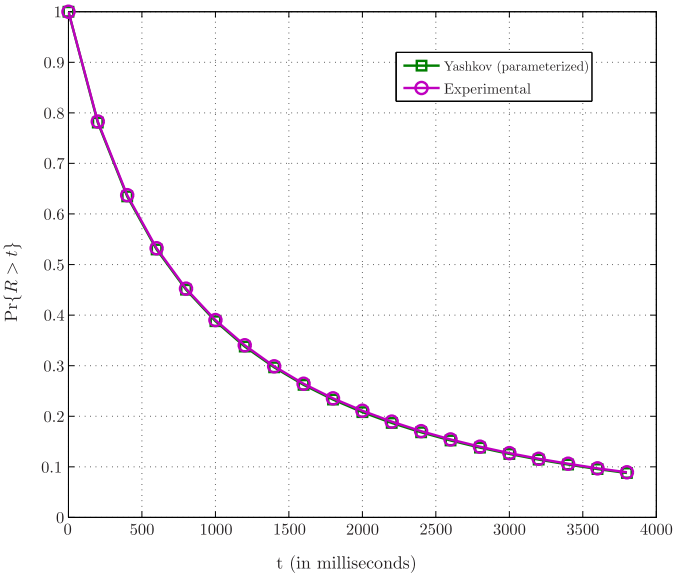

Fig. 2: Tail probabilities for transfer-time distribution: analytic vs. experimental results (for exponential job sizes).

Figure 2 shows again the model-based results are extremely close to those obtained in the test-lab setting.

B.3 Tail probabilities for non-exponential service times

For the case of non-exponential service times, the complete probability distribution of the sojourn time for the M/G/1-PS model is generally unknown, and is not insensitive to the distribution of the job sizes. Therefore, to validate the accuracy of the PS model (with effective service time) for predicting the transfer-time distributions in real systems, we have implemented the M/G/1-PS model in a discrete-event simulator. This enables us to obtain the 'exact' values of the tail probabilities of the sojourn times and compare them to the tail probabilities of the transfer times in the real system.

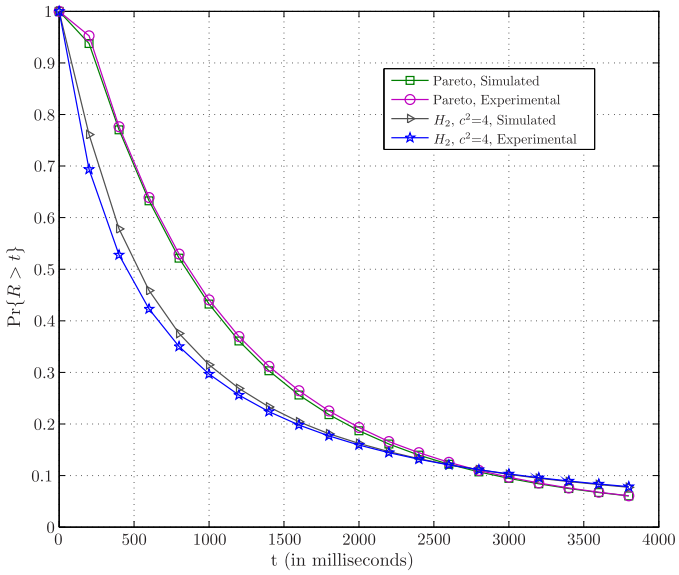

Fig. 3: Tail probabilities of the file-transfer times (mean file-size of 200kBytes) for hyper-exponential and Pareto-2 job-size distribution: analytic vs. experimental results.

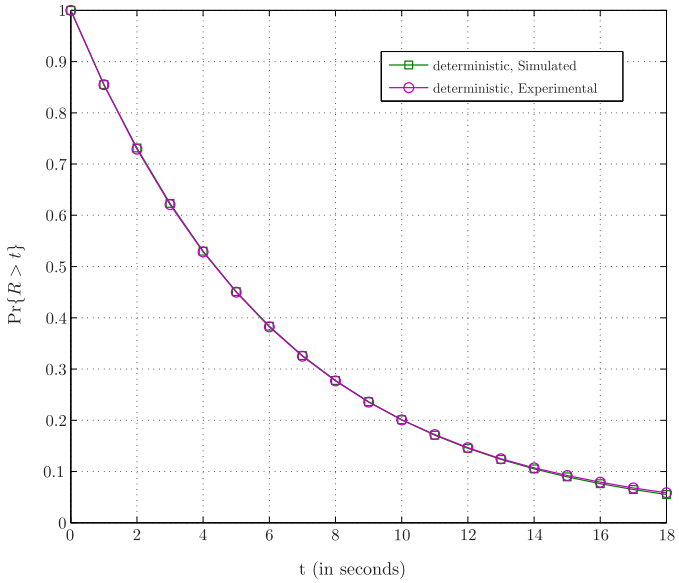

Fig. 4: Tail probabilities of the file-transfer times (constant file-size of 1MByte) for deterministic job-size distribution: analytic vs. experimental results.

Figure 3 and 4 show both the simulated ('exact') distribution and the experimentally obtained tail probabilities, and where the job-size distribution is varied as deterministic, two-phase hyper-exponential with squared coefficient of variation 4 and Pareto-2 (with infinite variance). The results in Figure 3 and 4 show again that the results based on the analytic model closely match the ones obtained in the test-lab setting. We also see that the influence of the file-size distribution on the distribution of the transfer times is nicely covered by the model.

B.4 Tail probabilities of number of flows

The results in (23) for the M/G/1-PS model suggest that the number of jobs in the real system has a (nearly) geometric probability distribution with parameter ρ_{eff}, and moreover, is nearly insensitive to the job-size distribution. To validate whether this is indeed the case for the real system, Figure 5 shows the tail probabilities of N on a log-scale for different job-size distributions (similar to the ones in Figure 1) for a system with $\beta_{eff} = 0.312$ and $\rho_{eff} = 0.78$.

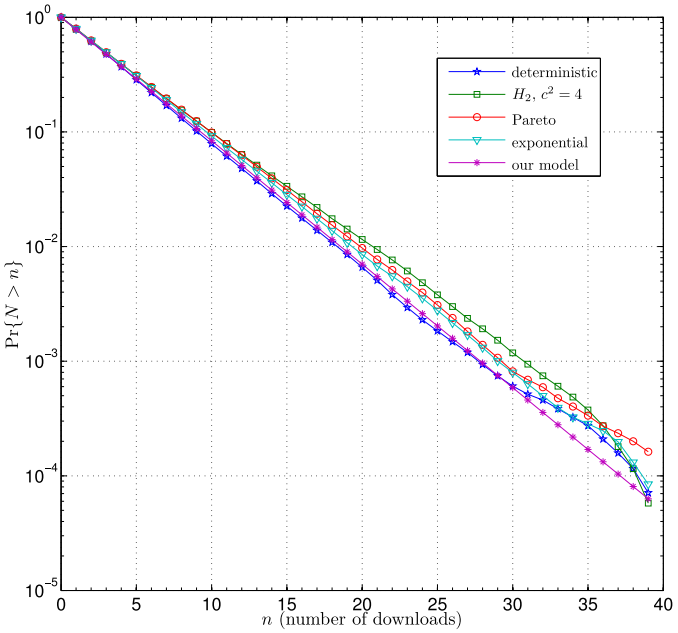

Fig. 5: Tail probabilities of number of jobs: analytic vs. experimental results.

The results in Figure 5 show again the model-based precistions closely match with the lab-test results. Moreover, the results also confirm the near-insensitivity of the number of jobs with respect to the file-size distribution.

V. Concluding remarks and further research

The main contribution of this paper is that it is an important first step towards bridging the gap between the powerful class of PS models for which a wealth of analytic results are known and complicated TCP-based communication networks on the other hand. This makes it possible for performance engineers to answer what-if questions under anticipated load scenarios.

The results suggest a number of challenges for follow-up research. First, the modeling approach presented in this paper was exemplified for the FTP/TCP/IP/WLAN protocol stack, but can be extended to other protocol stacks, including both wireless and wired networks. This requires in-depth analysis of the protocol details, similarly to the analysis in Section II. The development of similar models for other protocol stacks, and the evaluation of their accuracy, is a challenging topic for follow-up research. Currently, we are studying the parameterization for LTE networks. Second, a promising means to boost the performance of wireless and

wired networks is to make use of the fact that the densely populated areas are usually covered by a multitude of access networks. This phenomenon, often called Concurrent Access (CA), opens up tremendous possibilities for performance improvement by using multiple access networks at the same time. To fully exploit the possibilities of CA, smart algorithms are needed to efficiently split traffic among the different access networks. To this end, the ability to use parameterized PS-models to describe the flow-level performance at individual access networks (as discussed in this paper) is a crucial step in that direction.

VI. Acknowledgments

This work was performed within the project RRR (**R**ealisation of **R**eliable and Secure **R**esidential Sensor Platforms) of the Dutch program *IOP Generieke Communicatie*, number IGC1020, supported by the *Subsidieregeling Sterktes in Innovatie*. The contribution of R.D. van der Mei has been partially funded by the Dutch Ministry of Economic Affairs, project "Service Optimization and Quality" (IGC0820).

References

[1] S. Ben Fredj, T. Bonald, A. Proutiere, G. Régnié, and J. W. Roberts, "Statistical bandwidth sharing: a study of congestion at flow level," in *Proceedings of ACM SIGCOMM*, 2001, pp. 111–122.

[2] T. Bonald and J. W. Roberts, "Congestion at flow level and the impact of user behaviour," *Computer Networks*, vol. 42, no. 4, pp. 521–536, July 2003.

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

[4] G. J. Hoekstra and R. D. van der Mei, "Effective load for flow-level performance modelling of file transfers in wireless LANs," *Computer Communications*, vol. 33, no. 16, pp. 1972–1981, 2010.

[5] Y. Wu, C. Williamson, and J. Luo, "On processor sharing and its applications to cellular data network provisioning," *Performance Evaluation*, vol. 64, no. 9-12, pp. 892–908, 2007.

[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*, Budapest, 2001, pp. 101–112.

[7] T. Bonald and A. Proutière, "Wireless downlink data channels: user performance and cell dimensioning," in *Proceedings of ACM MobiCom*, 2003, pp. 339–352.

[8] S. Borst, "User-level performance of channel-aware scheduling algorithms in wireless data networks," *IEEE/ACM Transactions on Networking*, vol. 13, no. 3, pp. 636–647, 2005.

[9] A. Heindl and R. German, "The impact of backoff, EIFS, and beacons on the performance of IEEE 802.11 wireless LANs," in *IPDS '00: Proceedings of the* 4^{th} *International Computer Performance and Dependability Symposium*, Washington, DC, U.S.A., 2000, p. 103.

[10] J. Postel and J. Reynolds, "Standard for the transmission of IP datagrams over IEEE 802 networks," Internet Engineering Task Force, RFC 1042, February 1988.

[11] IEEE Standard 802.1H, "IEEE Standards for Local and Metropolitan Area Networks: Recommended Practice for Media Access Control (MAC) Bridging of Ethernet V2.0 in IEEE 802 Local Area Networks," 1995.

[12] J. Postel and J. Reynolds, "File transfer protocol (FTP)," Internet Engineering Task Force, RFC 959, October 1985.

[13] ProFTPD Project, "Professional FTP Daemon," November 2012, http://www.proftpd.org/.

[14] W. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*, 4th ed. New York, NY, U.S.A.: Addison-Wesley Professional, 1993.

[15] 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* 18^{th} *International Teletraffic Congress - ITC18*, Berlin, Germany, 2003, pp. 931–940.

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

[17] D. Miorandi, A. A. Kherani, and E. Altman, "A queueing model for HTTP traffic over IEEE 802.11 WLANs," *Computer Networks*, vol. 50, no. 1, pp. 63–79, 2006.

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

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

[20] S. F. Yashkov, "Processor-sharing queues: some progress in analysis," *Queueing Syst. Theory Appl.*, vol. 2, no. 1, pp. 1–17, Jun. 1987. [Online]. Available: http://dx.doi.org/10.1007/BF01182931

[21] 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, 2009.