33rd European Study Group with Industry
14 – 18 September 1998 pp. 11 – 26

Throughput of ADSL Modems

H.J. Broersma, J. Hurink
Universiteit Twente, Fac. Toegepaste Wiskunde,
Postbus 217, 7500 AE Enschede

N. Bruin
Rijksuniversiteit Leiden, Vakgroep SSOR,
Mekelweg 4, 2628 CD Delft

L.E. Meester
Technische Universiteit Delft

S.S. Op de Beek
KPN-Research

J. Westhuis
MARIN

This paper considers the throughput of ADSL (Asymmetric Digital Subscriber Line) modems, used for high speed data transmission over relatively unreliable connections, e.g., copper telephone wires. The modem technique uses an error correcting code and interleaving. The settings include a grouping factor S which affects the amount of data per code word, the number of redundant bytes per code word (R) and the interleave depth D. The influence of these parameters on both the effective data transmission rate and the resulting error rate in the received signal are determined for two error situations: random errors and bursts of errors. An approximate analysis for the random error case of the throughput of a TCP (Transport Control Protocol) connection using an ADSL modem shows that maximum throughput is obtained for the highest values of S and R.

1. Introduction

In providing data services, such as Internet access to customers, the voice band modems or even ISDN modems with maximum bit rates of 56 kbit/s and 64 kbit/s, respectively, form a bottle neck in the transmission. The recently standardised Asymmetric Digital Subscriber Line or ADSL modems [1] form a breakthrough in the access data rate by offering rates of several Mbit/s through the copper telephony infrastructure. High bit rate services, such as Video on Demand, Video conferencing or fast Internet can now be offered to and from a customer location. All these services have in common that the downstream data rate is much larger than in the upstream direction, which is reflected in the Asymmetric Digital Subscriber Line described below.

In this paper we examine the performance of TCP, the widely used Transport Control Protocol in the Internet, in case transmission is provided by an ADSL modem. In Section 1.2 the essential properties of TCP are described. Its performance is assessed by the TCP throughput: the net amount of correctly transmitted data per unit of time. TCP throughput in the Internet has been a subject of study for many years; its steady state performance has been analysed in [2, 3, 4]. The special case we study in this paper concerns the throughput of TCP over an ADSL link. Essentially, three specific ADSL aspects are of importance in this study:

1. the raw bit rate of the ADSL link,
2. the error correction techniques used in ADSL, and
3. the asymmetry of the data link.

The last issue will not be treated in this paper. Extensive study of the influence of the asymmetry of a data link on TCP throughput is described in [5]. The application of error correction techniques is inevitable in an ADSL modem, in order to obtain a reliable transmission link. The effective number of transmission errors in the ADSL data link depends on the adjustable settings in the data encoder. Section 2 treats the influence of the coding and the adjustable parameters, leading to an analysis whether and how transmission errors propagate to the decoded data stream. The influence of the remaining data transmission errors on the TCP throughput is described in Section 3. Finally, numerical results of the TCP throughput as a function of some typical ADSL settings are presented in Section 4.

1.1. ADSL

The Asymmetric Digital Subscriber Line (ADSL) is a modem technique that uses the copper telephony infrastructure to offer broadband data connections from a central office to subscribers. The asymmetry is due to the fact that generally, in high bandwidth applications, the data stream from the central office to the users is larger than the data stream in the opposite direction. By taking advantage of the asymmetry of the service, a higher downstream bandwidth can be achieved than with symmetrical modem techniques. The maximum raw bit rate typically is 8 Mbit/s from the Central Office (CO) to the customer (downstream direction), and 1 Mbit/s from the customer to the CO (upstream direction). The actual achieved bit rates strongly depend on the distance bridged by the modem pair and noise induced by other systems.

There are two different causes for bit errors in the local loop and for each of these ADSL takes different protective measures. The first cause of bit errors is the noise induced by other systems combined with the length of the copper cable. ADSL deals with this problem at start-up: it evaluates the line characteristics and estimates the induced noise on the transmission line. The initial transmission rate is optimised for the length and noise found. According to the

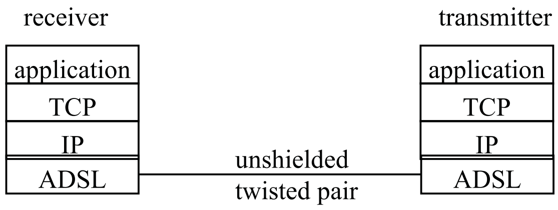

FIGURE 1. Typical protocol stack used in ADSL modems

ANSI ADSL standard [1], the maximum bit rate is determined such that the bit error ratio (BER) is less then 10^{-7}.

The second cause for bit errors is impulse noise. A common cause for impulse noise is the on and off switching of electrical equipment. This results in a short, but very strong disturbance of the line. Impulse noise can last from microseconds to several tens of milliseconds and results in a burst of bit errors. An interleaving technique increases the robustness of the ADSL modem for bit error bursts. Interleaving introduces an additional delay, typically between 10 and 60 milliseconds.

A typical protocol stack used in ADSL is shown in Figure 1.

1.2. TCP

The Transmission Control Protocol (TCP) provides a highly reliable stream of packets between transport layers on internet hosts by requiring acknowledgements from the receiving transport layer within a specified period of time. Furthermore, by providing sequence numbers, packets can be delivered in the order that they were sent. Error checking of each packet is provided by a check sum transmitted as part of the TCP header. TCP makes no assumption as to the reliability of the lower-level protocol. To achieve this level of control and reliability, a connection from the transport layer on one host to the other host must be set up before data can be transferred. Special handshake messages are defined in TCP for establishing and releasing a connection. In this paper, we will concentrate on the consequences of the flow control mechanisms in TCP. In TCP a so-called sliding window protocol is used. It allows the sender to transmit multiple packets before it stops and waits for an acknowledgement (ACK); the time until the acknowledgement of the first packet is received is called the round trip time (RTT). This leads to faster data transfer, since the sender does not have to stop and wait for an acknowledgement each time a packet is sent. Using special window update ACK messages, the receiver can inform the sender of the offered or advertised window size which depends on the receiver buffer filling. The sender can adjust its sending window to the minimum of the advertised window and the maximum window size (which is a fixed value which depends on the implementation of TCP). The process of sending packets and receiving ACKs is visualised in Figure 2. The most com-

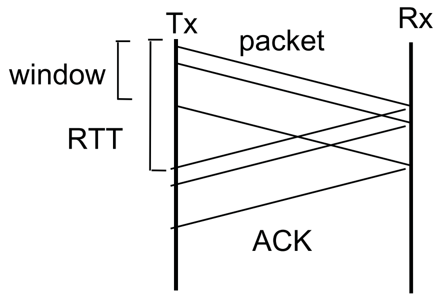

Figure 2. Data transfer during one Round Trip Time (RTT): a number of window packets is sent before the ACK of the first packet sent is received; Rx=receiver, Tx=sender.

mon TCP 'flavour' used in the internet today is TCP Reno. Four mechanisms are introduced in this variant in order to control the data transmission:

- Slow start. This begins by sending one packet and waiting for an acknowledgement. For each acknowledgement the sender receives, it injects two packets into the network. This leads to an exponential increase in the number of packets sent per RTT. The slow start phase ends when the receiver's advertised window is reached.
- Congestion avoidance. This mechanism is used to probe the network for available bandwidth by sending one additional packet for each RTT (up to the receivers advertised window). In the original slow start/congestion avoidance scheme, when the sending TCP detects packet loss (indicating congestion), it drops back into slow start until the packet sending rate is half the rate at which the loss was detected and then begins congestion avoidance.
- Fast retransmit and fast recovery. Fast retransmit reduces the time it takes a TCP sender to detect a single dropped packet. Rather than waiting for the retransmit timeout (RTO), the TCP sender can retransmit a packet if it receives three duplicate ACKs for the packet sent immediately before the lost packet. Fast recovery is closely related to fast retransmit: as mentioned before, when a sender retransmits a packet, it normally recovers by moving into a slow start phase followed by a congestion avoidance phase. If the sending TCP detects the packet loss using fast retransmit, however, fast recovery is used instead. Fast recovery halves the segment sending rate and begins congestion avoidance immediately, whitout falling back to slow start.

In our treatment we assume that the TCP receiver has an infinite processing capacity, i.e., the input buffer of the receiver never overflows. In Figure 3 a typical evolution of the window size as a function of time is shown for a typical TCP session. In our approach the slow start phase which is initated at connection start up, will not be taken into account.

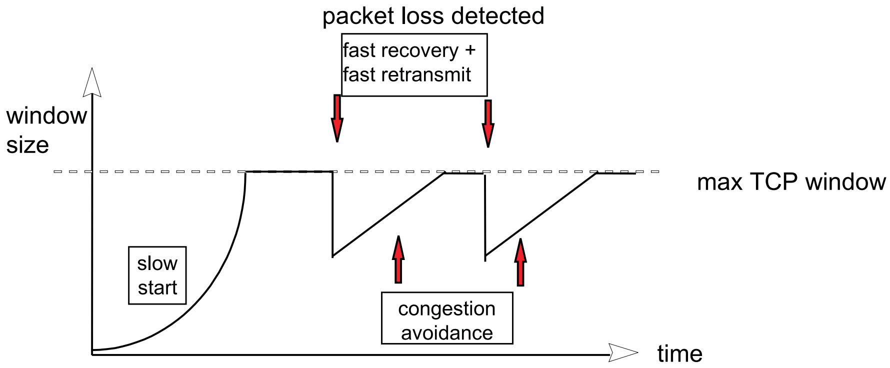

FIGURE 3. Typical behaviour of the TCP window size as a function of time

2. ADSL MODEM SPECIFICATION AND ERROR ANALYSIS

This section starts with a description of the ADSL modem in terms of its functional blocks. Not all of them are relevant to error propagation. In the error analysis that concludes this section only the relevant parts have been considered.

2.1. ADSL description

An ADSL modem can be divided into eight functional blocks (see Figure 4, modified from [1]). The first block groups bytes from a continuous byte stream (these bytes actually represent ATM cells) into a so-called multiplexer or Mux frame. This byte stream is supplied through one of two channels:

- The fast channel: data that is supplied through this channel follows a different functional path in the modem, resulting in a relatively short time delay of the byte stream. The drawback of this mode is that it has a relatively high error rate.
- The interleaved channel: this channel provides a relatively lower error rate, at the cost of a larger time delay.

The Mux-framer also adds a synchronization byte to the data in the Mux frame. Before actual transmission several Mux frames are assembled into one superframe. The second functional block computes Cyclic Redundancy Check (CRC) bits over a collection of frames that is to constitute one superframe. This check is used for internal evaluation only. The management system of the ADSL modem uses the CRC bits to keep track of the 'severely errored seconds' (SES) and 'errored seconds' (ES) statistics. The third functional block scrambles the data bytes in the framed byte stream to ensure that the resulting byte signal has a random character. In the case of the fast path, the bytes of a single Mux frame are coded directly into a so-called ADSL frame (functional block 4) using a Reed-Solomon coding algorithm. This Forward Error Correction

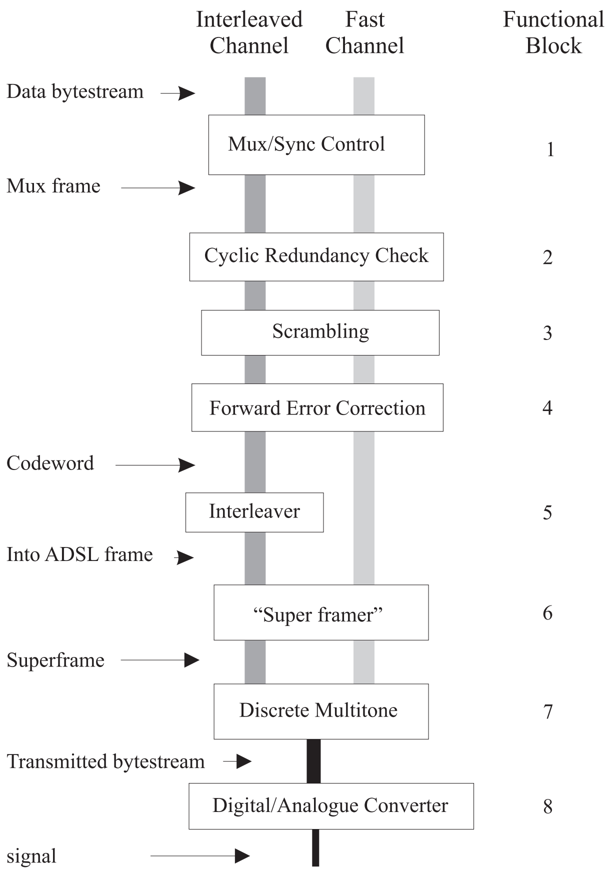

FIGURE 4. Functional blocks in an ADSL modem.

(FEC) coding ensures that when a small number of bytes in the code word is corrupted, the original Mux frame stored in the code word can still be recovered. The number of redundancy bytes per code word, added to the original signal for Forward Error Correction, can be adjusted and is denoted by R. The maximum number of error bytes that can be corrected increases as R increases (see Section 2.2).

When the interleaved buffer is used, multiple Mux frames are coded in a single code word and this code word is then interleaved with other code words from the interleaved buffer (fifth functional block). The main purpose of this is to spread the byte errors of a burst. We come back to this in Section 2.2.2. After the interleaving process, interleaved code words are partitioned into ADSL frames. The number of Mux frames per code word is denoted by S, the interleave depth is denoted by D; both these parameters can be adjusted.

In the sixth functional block, ADSL frames (which may originate from the fast or the interleaved path) are grouped, 68 at a time, into a superframe, and the originally computed CRC checksum is stored in the synchronization byte of the first ADSL frame of the superframe (functional block 7). Every ADSL frame in a superframe is modulated into a Discrete Multitone (DMT) symbol (seventh functional block). The total superframe consists of the original 68 frames plus an additional DMT symbol for synchronization purposes.

The Digital Analog Converter (last functional block) converts the stream into an electrical signal that is sent over the copper telephony infrastructure.

The question that we address is how the performance of the connection (both at ADSL and at TCP level) depends on the 'free' parameters R (number of additional FEC bytes per code word), S (number of Mux frames coded into a single code word) and D (interleave depth), considering both random and burst errors.

2.2. Error propagation at the ADSL level

First of all, we simplify the modelling by ignoring the scrambler and the conversion of the byte stream to DMT symbols and the digital-analog conversion. Furthermore, in the model we disregard the cyclic redundancy check since it is used for internal purposes only and has no influence on the behaviour or characteristics of the modem.

When the interleaved channel is used, S Mux frames are supplemented with R redundant bytes in order to create a code word. These code words are then interleaved to depth D. From a modelling point of view, we can identify the fast and the interleaved channels by assuming that the fast buffer equals the interleaved buffer with settings $S = 1$ and $D = 1$.

We shall use μ to denote the bandwidth of the physical link, in *bytes* per second; for computations we use $\mu = 2^{18} = 262144$, which corresponds to 2 Mbits/s. The baudrate, the number of superframes sent per second is nominally 4000, but because of the additional (the 69th) synchronization frame in the ADSL superframe, the actual baudrate is only $\frac{68}{69}$ of 4000. This means that it takes $\frac{68}{69} \cdot 0.25$ ms to send an ADSL frame, and so one obtains for the number of bytes in an ADSL frame:

$$N = \left\lceil \frac{68}{69} \cdot 0.25 \cdot 10^{-3} \mu \right\rceil \approx \lceil 2.46 \cdot 10^{-4} \mu \rceil = \alpha\mu;$$

the fixed factor α is introduced for notational convenience. The length of a code word in bytes, denoted by n, is a fundamental quantity in the analysis; it equals NS, so in fact n directly depends on the free parameter S. A code word contains information equivalent to $k = n - R$ bytes.

The fraction of actual data in an ADSL frame is determined by the number of synchronization bytes and redundant bytes from Forward Error Correction. In the case that S Mux frames are used to form a code word containing R redundant bytes, the final code word will be split into S ADSL frames, implying that the number of error correction bytes per ADSL frame equals R/S. The number of synchronization bytes in a Mux frame varies between 1 and 3; we will denote this unknown number as h and use $h = 2$ in computations. Thus the number of data bytes in an ADSL frame is $K = N - h - R/S$.

So a priori, without any considerations about errors, the efficiency of the ADSL modem, i.e., the number of data bytes transmitted divided by the actual number of bytes sent, given the parameters R, S and μ is:

$$\eta_{\mathrm{ADSL}} = \frac{K}{N} = 1 - \frac{h + \frac{R}{S}}{\alpha\mu}.$$

Random errors

In this case we assume that transmitted bytes are corrupted with probability p each, independently of the others. Whether interleaving is applied or not, the number of corrupted bytes has a Binomial distribution with parameters n and p. The probability that i bytes in a code word are corrupted is given by:

$$\binom{n}{i} p^i (1-p)^{n-i}.$$

The S Mux frames are encoded by an $(n, k, n-k+1)$ Reed-Solomon code over $\mathbb{F}_{256}$, which means that a data word of k bytes is coded as a code word of n bytes, such that distinct data words are coded as code words that differ in at least $n-k+1$ bytes. This implies that the original data word can still be recovered from a code word that has at most $e = \frac{1}{2}(n-k)$ damaged bytes.

Therefore the probability q, that the Mux frames coded in a Reed-Solomon code word cannot be recovered, given a byte error probability p, is given by

$$q = \mathrm{P}(\text{code word error}) = \sum_{i=e+1}^{n} \binom{n}{i} p^i (1-p)^{n-i}. \tag{1}$$

Random errors in the byte stream occurring with probability p result in a similar error process at code word level, but with (a smaller) probability q. An error in the code word stream, however, implies that *all* data bytes encoded into this code word are corrupted.

Bursts of errors

In the following we will analyse the effect of bursts of errors. We assume that a burst will lead to a complete corruption of L consecutive bytes in the byte stream. Furthermore, for sake of simplicity, we assume for the moment that the length L is a multiple of the interleave depth D: $L = cD$. Our aim is to get a relation between the length L of a burst and the (expected) number of lost code words (those that cannot be corrected by the Reed-Solomon code) in relation to the modem parameters. In this context, the main point of interest will be the interleaving.

As mentioned, code words are placed on the different layers of the interleaver and subsequently the bytes are read in such a way that each Dth byte belongs to the same layer (in Figure 5 the bytes are transmitted by reading columnwise). Thus a burst of length $L = cD$ leads to c consecutive corrupted bytes in each layer (see Figure 5).

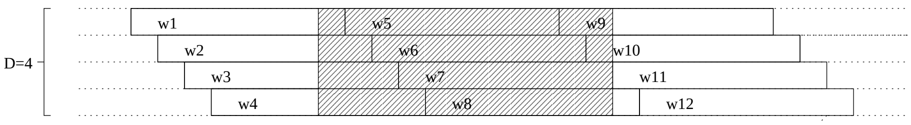

FIGURE 5. Sketch of a burst

In order to determine the number of lost code words, we consider the layers separately. Although the position of the first corrupted byte within a code word differs between the layers, the expected number of corrupted code words will be the same for each layer.

The following lemma analyses the effect of a burst for one layer of the interleaver.

LEMMA 1 *Consider a byte stream of code words of length n which are coded by a Reed-Solomon code with $R = 2e$ redundant bytes. The expected number of corrupted code words for a burst of length c is given by:*

$$E = \begin{cases} 0 & \text{if } c \leq e, \\ 1 + \frac{c-2e-1}{n} & \text{if } c \geq e+1, \end{cases}$$

PROOF: In the case $c \leq e$, each code word has at most e corrupted bytes and, thus, each code word can be corrected by the Reed-Solomon code.

To analyse the case $c \geq e+1$, consider the position t of the first byte of the burst within a code word ($t = k$ means that the first destroyed byte is the kth byte of some code word). Let cw_1 denote the code word containing the first byte of the burst and let $cw_2, cw_3, \ldots$ denote the following code words.

To determine the total number of corrupted code words in dependence of t, let $b := \lceil \frac{c-e-1}{n} \rceil$. We will show that code words $cw_2, \ldots, cw_b$ are always destroyed, the possibility of correcting cw_1 and cw_{b+1} depends on t, and that the remaining code words all can be corrected.

Obviously, code word cw_1 can not be corrected if and only if $t \in \{1, \ldots, n-e\}$.

Furthermore, code words cw_i with $i \geq 2$ can not be corrected if and only if

$$t + c - 1 \geq (i-1)n + e + 1.$$

So, since

$$(b-1)n + e + 1 \leq \frac{c - e - 1 + n - 1}{n} n - n + e + 1 = c - 1 < t + c - 1$$

we may conclude that the code words $cw_2, \ldots, cw_b$ can not be corrected.

Since $1 \le \frac{c-e-1}{n}n - c + e + 2 \le bn - c + e + 2 \le \frac{c-e-1+n-1}{n}n - c + e + 2 = n$ we may conclude that code word cw_{b+1} can not be corrected if and only if $t \in \{bn - c + e + 2, \ldots, n\}$.

Finally, since $(b+1)n + e + 1 \ge \frac{c-e-1}{n}n + n + e + 1 = c + n > c + t - 1$, code words $cw_{b+2}, cw_{b+3}, \ldots$ are all correctable.

The expected number of corrupted code words now can be obtained by considering all possible values for t. Since t is uniformly distributed on $\{1, \ldots, n\}$ we get the following expected number of corrupted code words:

$$\begin{aligned} E &= \tfrac{n-e}{n} + b - 1 + \tfrac{n-(bn-c+e+2)+1}{n} \\ &= b - 1 + \tfrac{n(2-b)+c-2e-1}{n} = 1 + \tfrac{c-2e-1}{n} \end{aligned}$$

□ From the lemma, we get for the interleaver as a whole:

THEOREM 2 *Consider an ADSL modem using interleave depth D and code words of length n coded by a Reed-Solomon code with $R = 2e$ redundant bytes. A burst of length $L = cD$ ($c \in \mathbb{N}$) will result in an expected number of*

$$E = \begin{cases} 0 & \text{if } c \le e, \\ D(1 + \frac{c-2e-1}{n}) & \text{if } c \ge e+1, \end{cases}$$

not correctable code words.

So the maximum length of a burst that can be tackled by an interleaver with interleave depth D and a Reed-Solomon code with $R = 2e$ redundant bytes is eD bytes.

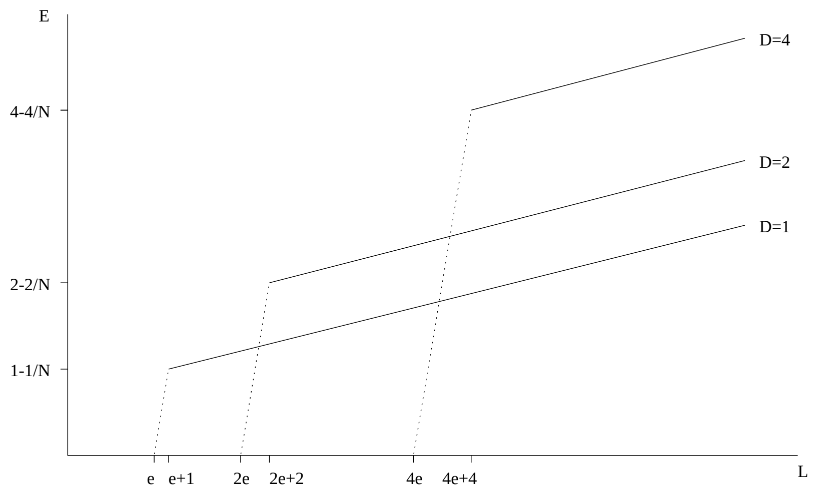

FIGURE 6. Expected number of corrupted code words for different values of D.

In Figure 6 we have sketched the expected number of corrupted code words in relation to the burst length for some values of D. It is interesting to see that a higher interleave depth, besides the positive effect that longer burst can be handled without getting any corrupted code words, also has a negative effect: if a burst is too long to disappear completely, the expected number of corrupted code words increases with the interleave depth.

Up to now, we have focused on the expected number of corrupted code words. Of course, also the distribution is of interest, so we try to say something about the range of values that may occur. As a direct consequence of the proof of Lemma 1 we can state that a burst of length $L = cD$ leads to $b-1$, b, or $b+1$ corrupted code words per layer, where $b = \lceil \frac{c-e-1}{n} \rceil$. Furthermore, it is easy to see that this number will be equal to $b-1$ or b if $(c \mod n) \leq 2e+1$ and equal to b or $b+1$ if $(c \mod n) \geq 2e+2$. As an immediate consequence we get:

LEMMA 3 *Consider an ADSL modem using interleave depth D and code words of length n coded by a Reed-Solomon code with $R = 2e$ redundant bytes. For a burst of length $L = cD$ ($c \in \mathbb{N}$) the number F of corrupted code words belongs to the set*

$$\{mD, mD+1, \ldots, (m+1)D\} \text{ with } m = \begin{cases} b-1 & \text{if } (c \mod n) \leq 2e+1 \\ b & \text{if } (c \mod n) \geq 2e+2. \end{cases}$$

By a more detailed analysis of the dependences between the different layers of the interleaver, the possible values for F may be reduced further.

The above considerations are based on the assumption that the burst length is a multiple of the interleave depth D. However, the stated results can be adapted in a straightforward manner to general burst length $L = cD + a$ with $c \in \mathbb{N}$ and $a \in \{0, \ldots, D-1\}$ since such a burst leads to a layers with a burst of length $c+1$ and $D-a$ layers with a burst of length c.

3. EFFECT ON TCP THROUGHPUT

In this part we consider the performance of an ADSL modem link where actual data transmission is controlled by a Transport Control Protocol (TCP). We first consider the occurrence of errors as seen by TCP.

The data stream that is transformed and transmitted by the ADSL modem in fact is a stream of *packets* generated by TCP. Let g be the number of bytes in a packet. The transmitting modem chops the stream in pieces of k bytes, the data words, and, at the other end, the receiver reassembles the TCP packet stream from them. We assume that an error in a code word results in the detection of an error by TCP in any and all packets that overlap the corresponding data word. Clearly, a code word error may lead to one, two or several packet errors, depending on k, g, and whether there is any alignment. We speak of alignment when there are (periodic) instances in the byte stream where code word and packet boundaries coincide.

Now consider the loss of a randomly chosen code word. By simple counting arguments the following probabilities are easily obtained. If $k \leq g$ and there is no alignment:

$$\mathrm{P}(\text{one packet lost}) = 1 - \frac{k}{g}, \qquad \mathrm{P}(\text{two packets lost}) = \frac{k}{g};$$

if there is aligment:

$$\mathrm{P}(\text{one packet lost}) = 1 - \frac{k}{g}(1 - \frac{l}{k}), \qquad \mathrm{P}(\text{two packets lost}) = \frac{k}{g}(1 - \frac{l}{k}),$$

where $l = \mathrm{lcm}(k, g)$, the least common multiple of k and g.

If $k > g$ the number of lost packets is

$$\left\lceil \frac{k}{g} \right\rceil \quad \text{or} \quad \left\lceil \frac{k}{g} \right\rceil + 1.$$

If there is no alignment, it is always the latter. If there is:

$$\mathrm{P}\left(\left\lceil \frac{k}{g} \right\rceil \text{ packets lost}\right) = \frac{2l}{g}, \qquad \mathrm{P}\left(\left\lceil \frac{k}{g} \right\rceil + 1 \text{ packets lost}\right) = 1 - \frac{2l}{g}.$$

From these results it is concluded that random (code word) errors appear as (usually small) clusters of errors in the packet stream.

Now we consider the effect on the throughput using the Reno version of TCP in the *congestion avoidance phase*. It is convenient to consider time in a sequence of intervals of length equal to the *Round Trip Time* (RTT). TCP controls transmission by means of its adaptable *window size*, W: the amount of data to be released for transmission in the next RTT interval. The receiving TCP sends back acknowledgments for the undamaged reception of packets; damaged or lost packets are detected through the ackowledgments. In the congestion avoidance phase, if all packets sent in the previous RTT are received undamaged, the window size is increased by 1 packet; if not, the window size is halved and missing packets are retransmitted. So loss in throughput is caused largely by the reduction of the window size.

Random multiple packet loss and severe error situations result in time-outs, which cause TCP to start again with a *slow start phase* in which the window size is set to 1 packet. From the above it is clear, however, that this is an important issue, since random (single) code word errors that lead to multiple packet losses are by no means exceptional. Unfortunately, no generally valid accurate description of how TCP handles this was available, and this could not be analysed.

This also implies that the effect of error bursts on the TCP throughput can only be analysed in so far as they lead to single packet losses, though, as is clear from previous sections, this is a rare occurrence. Interleaving, however, also affects the RTT and, through the RTT, the effective throughput.

Assume that a single TCP packet is coded into a single code word (of size n) . The round trip time now consists of a fixed part, RTT_0, which consists of the time needed to do the coding and decoding and the time to send the acknowledgement, and the time needed to actually transmit the bytes of the code word. This is roughly nD/μ seconds. So,

$$\mathrm{RTT} = \mathrm{RTT}_0 + \frac{nD}{\mu}.$$

The maximum number of bytes transmitted in one RTT is therefore

$$W_{\max} = \mu \mathrm{RTT}.$$

We assume that this maximum window size is used until an error occurs. Subsequently, $W_{\max}/2n$ error-free RTT periods are needed to re-attain the maximum window size. During this phase on average an extra $W_{\max}/4$ bytes per RTT could have been transmitted had the error not occurred. So the error results in a loss of transmission capacity of $W_{\max}^2/8n$. Assuming that each error results in a loss this size leads to an upper bound on the total loss, as the additional loss due to an error during a recovery phase is less than the loss when $W = W_{\max}$.

If q, the probability of a code word error (see also Section 2.2) is sufficiently small, then code word errors will be so infrequent that they nearly always occur when the window size is completely recovered from the previous error. Then we expect, using $W_{\max} = \mu\mathrm{RTT}$, for the effective bandwidth:

$$\mu_{\text{eff}} = \mu \left(1 - q\frac{\mu^2\,\mathrm{RTT}^2}{8n}\right).$$

Of course, we are interested in the number of *data* bytes we expect to transmit per second, in relation to μ. Using the result obtained in Section 2.2 for the efficiency η_{ADSL}, we obtain

$$\eta_{\mathrm{TCP}} = \frac{\mu_{\text{eff}}}{\mu}\eta_{\mathrm{ADSL}} = \left(1 - \frac{q\mu^2\,\mathrm{RTT}^2}{8n}\right)\left(1 - \frac{h + \frac{R}{S}}{\alpha\mu}\right). \tag{2}$$

If, for argument's sake, we assume that RTT_0 is negligible, then we can simplify this expression even further to

$$\eta_{\mathrm{TCP}} = \left(1 - \frac{1}{8}qnD^2\right)\left(1 - \frac{h + \frac{R}{S}}{\alpha\mu}\right).$$

where q depends on n and R, and n on μ and S.

	$R = 0$	2	4	6	8	10	12	14	16
$S = 1$	0.489	0.984	1.000	1.000	1.000	1.000	1.000	1.000	1.000
$S = 2$	-0.981	0.874	0.995	1.000	1.000	1.000	1.000	1.000	1.000
$S = 4$	-6.441	0.075	0.922	0.995	1.000	1.000	1.000	1.000	1.000
$S = 8$	-25.4	-5.26	-0.037	0.869	0.987	0.999	1.000	1.000	1.000
$S = 16$	-83.1	-35.3	-10.4	-1.8	0.44	0.905	0.986	0.998	1.000

TABLE 1. μ_{eff}/μ for $p = 0.001$, $\mu = 262144$ bytes/s, $D = 1$.

4. Results and Discussion

Using formula (2) for η_{TCP}, we can calculate the expected utilisation of the total available bandwidth during bulk data TCP-transmission over an ADSL connection with random errors occurring at some fixed rate p.

Table 1 shows the values of μ_{eff}/μ for various parameter settings. This quantity measures the throughput in relation to the throughput that would have been attained if the same settings were used over an error-free connection ($p = 0$). The negative values correspond to cases where the word error rate is so high that the assumption that word errors occur only if TCP has recovered from the previous word error, is too far from the truth.

We see that higher settings of R and lower settings of S result in an error correction that is so good that the probability of byte errors resulting in word errors is so small that their effect is smaller than the precision of the table. This table would suggest that setting R very high has no negative effect. This is misleading, as Table 2 shows. The design of ADSL modems implies that the transmission speed of data *together with* redundancy and overhead is fixed. Thus, increasing the redundancy reduces the bandwidth available to data. Note the combinations $(S, R) = (1, 2)$, $(2, 6)$ and $(4, 12)$, which have about the same η_{TCP}.

	$R = 0$	2	4	6	8	10	12	14	16
$S = 1$	0.474	0.923	0.907	0.877	0.846	0.815	0.785	0.754	0.723
$S = 2$	*	0.834	0.933	0.923	0.908	0.892	0.877	0.862	0.846
$S = 4$	*	0.0717	0.879	0.941	0.938	0.931	0.923	0.915	0.908
$S = 8$	*	*	*	0.832	0.941	0.949	0.946	0.942	0.938
$S = 16$	*	*	*	*	0.423	0.869	0.945	0.954	0.954

Table 2. η_{TCP} for $p = 0.001$, $\mu = 262144$ bytes/s, $D = 1$.

As we see, for $S = 16$, $R = 14, 16$, we have the highest value $\eta_{\mathrm{TCP}} = 0.954$. In fact, calculations to a higher precision yield that for $R = 14$, η_{TCP} is strictly larger. Thus, for the presented p and μ, $(D, S, R) = (1, 16, 14)$ gives the best expected throughput.

Since we do not consider bursts of errors in this model, interleaving will have no positive effect. On the other hand, interleaving does increase RTT and thus increases the penalty on a word error. Table 3 shows that this effect can be quite dramatic. The value for $(D, S, R) = (64, 16, 16)$ is not realistic; the RTT will be so large in this case, that the recovery periods of word errors will overlap significantly.

This discussion shows that the interleaver should only be used if we expect a significant reduction of the number of corrupted code words. Since such a reduction is only achieved if bursts can be removed completely (otherwise the use of the interleaver will even increase the number of corrupted code words slightly), the probability that a burst of length smaller than eD (see Corollary 1) occurs has to be significantly larger than 0 to get a positive effect of the use of the interleaver at depth D.

The original question was about throughput during a long transmission. That means that the penalty on large values for RTT is only felt through the increased recovery time of the window size. In practice there is another unfortunate side-effect of large RTTs, namely a lower responsiveness of the system. For interactive applications, high responsiveness is generally highly desirable. As we saw before, assuming that packets coincide with code words,

$$\mathrm{RTT} = \mathrm{RTT}_0 + \alpha S D,$$

where $\alpha \approx 2.46 \cdot 10^{-4}$ sec.

The assumption that $\mathrm{RTT}_0 = 0$ is questionable. This off-set value of the round trip time is largely determined by the so-called packetizing delay, and can be considered fixed in the back-to-back modem pair set-up of this paper.

As an abstraction, it might be desirable to view the ADSL connection as a whole as a byte stream channel. This channel will have a byte error rate associated to it, which is exactly q as calculated. Note however, that byte errors on this channel will definitely not be uncorrelated, errors will appear in groups corresponding to data words.

	$R = 0$	2	4	6	8	10	12	14	16
$S = 1$	*	*	*	0.856	0.846	0.815	0.785	0.754	0.723
$S = 2$	*	*	*	0.264	0.891	0.892	0.877	0.862	0.846
$S = 4$	*	*	*	*	*	0.888	0.922	0.915	0.908
$S = 8$	*	*	*	*	*	*	0.620	0.921	0.937
$S = 16$	*	*	*	*	*	*	*	*	0.169

TABLE 3. η_{TCP} for $p = 0.001$, $\mu = 262144$ bytes/s, $D = 64$.

A. List of symbols

- μ — Bandwidth in bytes/s: number of bytes per second that are transmitted over the physical link.
- N — Number of bytes in an ADSL frame.
- n — Number of bytes in a code word.
- K — Number of data bytes in an ADSL frame.
- k — Number of data bytes in a code word.
- R — Number of redundancy bytes added to a code word for error correction. Admissible values for R are $(0, 2, 4, 6, 8, 10, 12, 14, 16)$.
- D — Interleave depth. Admissible values for D are $(1, 2, 4, 8, 16, 32, 64)$.
- S — Number of Mux frames to be encoded in one code word. Admissible values for S are $(1, 2, 4, 8, 16)$.
- e — Number of errors that can be corrected in a code word
- L — Length of a burst of errors in bytes.
- g — Number of bytes in a TCP packet.
- η_{ADSL} — Number of data bytes divided by the number of transmitted bytes through the ADSL modem.
- η_{TCP} — Long time average fraction of data byte rate and the maximum byte rate (using TCP and an ADSL modem).

References

1. *Asymmetric Digital Subscriber Line (ADSL) Metallic Interface*, T1E1.4/97-007R6, ANSI, 26 september, 1997
2. Performance Analysis of Window-based Flow Control using TCP/IP, T.V. Lakshman and U. Madhow, *High Performance networking V*, Vol. C-26, 135–149, 1994.
3. Modelling TCP Throughput: a Simple Model and its Empirical Validation, J. Padhye, V. Firoio, D. Towsley, J. Kurose, *Proceedings of SIGCOMM 98.*
4. The Macroscopic behavior of the TCP Congestion Avoidance algorithm, M. Mathis, J. Semke, J. Mahdavi, and T. Ott, *Computer Communications Review*, 27(3), 1997.
5. The Effect of Asymmetry on TCP Performance, H. Balakrishan, V. Padmanabhan, and R. Katz, *Proc. 3rd ACM/IEEE Mobicom*, september 1997.