An end to end simulator for performance comparison of different TCP schemes over UTRA channels

Eugenio Costamagna, Lorenzo Favalli, Matteo Lanati and Francesco Tarantola
Università di Pavia, Dipartimento di Elettronica, Via Ferrata 1, 27100 Pavia, Italy
e-mail {name.surname}@unipv.it

***Abstract*-One of the targets of third generation mobile radio systems and among these of the Universal Mobile Telecommunication System (UMTS) is the provision of Internet applications to mobile users. Building an end-to-end TCP connection involves the delicate equilibrium of flow control parameters which is severely affected by the channel conditions in the radio part and by the retransmission procedures used to improve reliability. As different versions of TCP have been introduced through the years, in this paper we report a comparison among some of these versions implementing an end-to-end TCP connection using Network Simulator (NS-2) with a final segment emulating the behavior of the UMTS system (UTRAN-FDD mode in the current version) by means of a Hidden Markov model that generates successive error events at the PDU level.**

***Keywords*-UMTS, TCP, performance.**

1. INTRODUCTION

In these years, the IP based stack of protocols is replacing all previous technologies in fixed networks, even in traditional telephone networks. In this path, IP and related protocols have grown to support the wide variety of services and mainly to provide some guaranteed quality of service. Mobile radio networks are part of this same process although the impact of mobility and the peculiarities of the radio channel require a careful analysis of the actual capabilities of the IP and TCP protocols. On its side, this approach makes a unique platform for the delivery of high-speed data and multimedia services. The odds are related to the fact that Internet protocols have not been built to tackle the two main issues of wide area wireless networks: mobility and information errors due to adverse channel conditions. The third generation partnership project (3GPP) has developed an evolution path from the second-generation (2G) Global System for Mobile Communications (GSM), through *General Packet Radio Service* (GPRS), to *UMTS Release 1999* (UMTS R99), and *UMTS Release 2000* (UMTS R00) that leads to a full integration with the world of IP-based networks. In particular UMTS Release 2000 has been split up into Releases 4 and 5. Release 4 introduces a next-generation network architecture for the *circuit-switched* (CS) domain. Release 5 introduces the *IP multimedia* (IM) subsystem on top of the *packet-switched* (PS) domain. These changes may well be tracked in the many documents provided by the 3GPP itself [1], the UMTS forum [2], and a large number of papers and books (see [3],…,[8]) for example. For what the radio link is concerned, a lot of research has been made in these years to determine how the performance of the TCP protocol, responsible for quality of service in the Internet, are affected by the higher latencies of a radio channel [6][8][9]. The focus of the paper is the determination of some quality of service (QoS) parameters in a IP+wireless system comparing different TCP control schemes. The general conclusions appear to be rather similar to those published in [8], although a direct comparison cannot be made since the simulation conditions start from opposite hypothesis. Furthermore, comparison among different TCP schemes allows to identify the most critical feature of the TCP flow control. The wireless system is an implementation of the ETSI standard UTRA-FDD which has been first modeled in detail "at the chip level" considering transmission to be carried out on a Dedicated channel (DTCH). Since the performances are affected by the interaction of the TCP parameters and the algorithms at the RLC layer of the UMTS protocol stack, a short review of these protocols is given in sections 2 and 3. The simulator is described in section 4. In order to be able to perform end-to-end simulations and to evaluate the performances at the TCP level, the lower layer simulations need to be interfaced with a packet level simulator. For this purpose, we chose to use *Network Simulator 2* (NS2), which is freely available and widely used in the research community. To allow fast simulations, the link level simulator has been modeled as a hidden Markov model to generate error events at the radio block level with the proper statistics. Details of the model and simulation results are in section 5.

2. UMTS PROTOCOL STACK

The UMTS system is a form of *wideband CDMA* and can be categorized into pure wideband CDMA and wideband time division (TD) CDMA. Pure wideband CDMA uses frequency division duplex (FDD) to organize the uplink and downlink transmissions, while wideband TD-CDMA uses time division duplex (TDD). TDD mode is well-suited for environments with high traffic density and applications with highly asymmetric traffic, while FDD mode is advantageous for applications in public macro- and micro-cell environments, nonetheless FDD mode is the one commonly

This work supported under FIRB project TANGO of the Italian Ministry of Education. Project no. 123456.

in place for commercial use. Frames of 15 slots multiplexing data and signaling information are built and transmitted at a chip rate of 3.84 Mchip/sec. In FDD mode wideband CDMA, the resource units only include codes in a frame: to enhance the flexibility of resource allocation, orthogonal-variable-spreading-factor (OVSF) is used [3]. A mobile terminal can have multiple codes and the OVSF of each code can be variable. Since the final chip rate is fixed, rate adaptation for the different applications is needed and includes options for the error coding techniques: the actual number of bits transmitted in a frame then varies. At the radio interface, the IP sits on top of the Radio Resource Control (RRC) sublayer which is responsible of monitoring the link status. The actual treatment of the data to be transmitted starts with the Radio Link Control that provides the necessary segmentation and reassembly functions and handles the retransmission of errored PDUs. The complex structure of logical and physical channels and the coding options is then handled at the MAC and L1 layers.

The basic RLC scheme consists of the transmitter sending polls to the receiver, and the receiver sending status report back to the transmitter indicating which of the RLC-PDU have been successfully received and which have not.

The RLC provides three different data transfer services:

1. *Transparent mode (TM),*
2. *Unacknowledged Mode (UM),*
3. *Acknowledged Mode (AM)*

The first two types of services are suitable for paging or streaming applications where error-free transmission must be traded with delay considerations. The AM transmits higher layer packets ensuring the verification of the correctness of the transmission. The error recovery scheme uses type I hybrid ARQ. There are three types of hybrid ARQ schemes, stop-and-wait (S&W), go-back-N (GBN) and selective repeat (SR). RLC can use any one of the retransmission schemes. In AM, the transmission of the acknowledgements is based on a poll mechanism [11][12] in which the sender side explicitly asks the receiver to ack the PDUs transmitted via STATUS report PDUs. The transmission window and acknowledgement format can be configured appropriately: The frequency of the POLL-STATUS exchange obviously strongly affects the compromise between the delay performance and the overhead of the transmission protocol. The AM mode functionalities are summarized in Figure 1 [11]. Note that PDUs are acknowledged in the lower part of the layer, while information from the user data is handled in the upper part where piggyback information is extracted. In view of transmission of TCP packets this is the point where acks to the TCP sender must be seen.

At the MAC sub layer, mapping is performed onto the transport channels thus determining the (eventual) multiplexing scheme and the error control procedures to be used. This task is performed at Layer 1 together with all the other transmission functionalities.

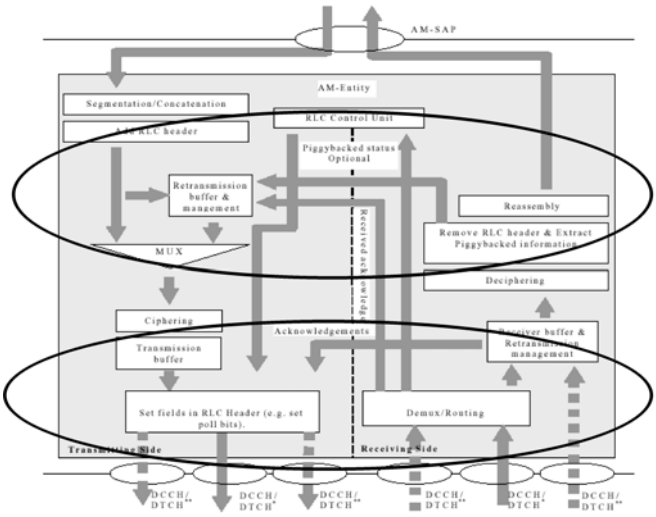

Fig. 1. Block diagram for the Acknowledged Mode.

3. TCP CARACTERISTICS

The basic connection control mechanism of the Transport Control Protocol is well known as well as its motivation and origins. Consequently here we only recall the main differences among the versions of the protocol that we have compared which are those implemented in the version of Network Simulator 2 (NS2) used in our simulations [13].

Four versions of TCP have been considered: Tahoe, Reno, New Reno, and Selective Acknowledgement. The first was introduced in 1988 [14], and already includes the basic congestion control algorithms used today in IP networks: the data is divided into segments and the transmission is adjusted considering a congestion window at the sender side and a receiver window at the destination. The last three versions have basically the same features of Tahoe, but introduce new tricks to better manage the available resources.

When a packet loss occurs, two typical causes can be identified: a timer expiry, when a packet takes too much time to reach the destination, and a segment loss during the transmission. Tahoe is able to distinguish these two events, but it uses the same trick in both cases: the slow start, i.e., the transmission window is set to 1 packet. The Reno version, instead, considers a duplicate ack a signal of channel error, and not congestion: in this last case, in fact, ack should not arrive to the source, while in case of errors, some ack packets could be delivered to the source, informing that a certain amount of data have not correctly reached the destination node. To better solve an erratic delivery, Reno uses the fast recovery algorithm, instead of the slow start one [15]: after three duplicate acks, fast retransmit is applied, without waiting for timer expiry, and avoiding the slow start phase. The threshold is halved, but the congestion window becomes equal to it and is incremented by 3 MSS. Congestion window is increased by a MSS after each new duplicate ack, until a

segment loss confirmation. In this case congestion avoidance is applied and the window becomes equal to the threshold.

The third TCP version considered uses the selective ack to signal to the sender that a group of non adjacent segments is arrived to the destination, which is waiting for the retransmission to complete the data flow [16].

New Reno [17] introduces partial ack to verify only segment parts, sent during the fast retransmit and fast recovery phases: this function permits to solve the multiple packet losses in a window without selective ack, but only a single lost segment at a time can be sent. In detail, when a non duplicate ack is received, congestion avoidance is prevented, and, in case of different segment losses, the retransmission of the first not confirmed segment is executed. The procedure continues until all the data are confirmed, and in this case congestion avoidance is restarted.

4. LINK AND MAC LAYER SIMULATIONS

Simulation of bit level errors is usually a lengthy process also with modern computing facilities. These times are not compatible with packet level simulations where a single packet is made of a few thousands bits. For this reason, it is common practice to assume that the channel behaves as a simple Binary Symmetric Channel (BSC) at the packet level (e.g. [8]). This is true only for very low error rates or for long interleaving lengths, which may not be compatible with the time outs of the flow control mechanisms. In other works a Gilbert [18] channel is used with generic assumptions on the state probabilities [19].

For this reason, we decided to implement a 'chip level' simulation of a UMTS radio channel complying with the specifications of the standard. To keep the resource assignment problem out of the simulator, the current version of the simulator implements and multiplexes the data structures of the DTCH/DCCH to build a DPDCH, which is then multiplexed with the DPCCH. A rate 1/3 turbo coding (three decoding iterations) is used to protect the information that flows along a 3GPP multipath channel. The interference from other cells is simply considered as an additional interference term. In the receiver we assume a three fingers rake receiver. Three different connection speeds have been taken into account: 64, 144 and 384 kbps with fixed length PDUs of 40 bytes two of which are header bytes and the others represent the payload (see Table 1).

The successive step has been to model the error sequences and the retransmission procedure by means of hidden Markov models (HMM) and the inclusion of such models in an IP network simulator (NS2) to simulate an end to end network (see next section). The retransmission process has been statistically characterized and modeled using the Baum-Welch algorithm [21]. It has been found that at the target BER (bit error rate) of 10^{-6} the transmission chain is rather good in randomizing the residual error process at the PDU level [20] and the ARQ protocol may be characterized using a simple two states Markov model. Note that this does not hold for the error process at the bit level and also may not be considered a valid assumption if the bit error rate approaches 10^{-4} at which the number of states dramatically increases and at least 8 states are necessary in agreement with [22].

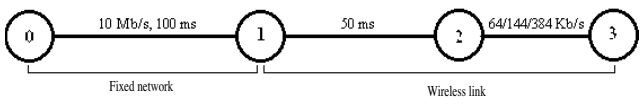

Fig. 2. Network topology. Note how Node B is splitted following the functionalities of the lower layers.

Table 1. Simulation parameters.

Source Rate [Kbps]	PDUs per block	block size [Byte]	Transm. Time Interval	Spread Factor	Blocks in DTCH
64	4	160	20 ms	16	1
144	9	360	20 ms	8	2
384	12	480	10 ms	4	4

5. RESULTS FOR TCP TRANSMISSION

Network topology has been set to be very simple (see), and only consists of a 'destination' node connected via a link with desired UMTS-like capacity and error model. Delay on the wireless link is due to segmentation and coding as propagation delay is negligible. The HMM step has been necessary to achieve good speed performance for the end to end simulations since simulation of 10^8 bits (which only allow for an average number of packets at least two orders of magnitude lower) takes a whole night on a 2GHz, 512 MB RAM P4 computer. We must underline that no resource allocation protocol has been implemented at the UMTS interface, so a single connection is considered and the transmission channel is always available when necessary.

The delays introduced are then only due to retransmissions caused by radio channel errors, and are not connected to buffering at the packet scheduler.

Some modifications of NS2 have been introduced to simulate the wireless link: a new "radio agent" has been created, which simulate the transmission over a wireless channel. TCP packets coming from the wired channel, after arriving on the UMTS gateway, have to be divided into PDU segments: in our simulations the size of TCP packets is set to 576 bytes, so 15 PDUs for each packet have been used. On UMTS side, a transmission window of 15 PDUs has been considered: a bigger window size could create problems to the TCP "self clocking" mechanism of the source. Transmission of a new packet is triggered by the arrival of an ack to avoid network congestion, but a large window delays retransmission of errors and generation of acks. Consequently TCP stops sending data, the Tx-window cannot shift and lost PDUs cannot be recovered: the situation is stalled. This is the reason for choosing a smaller window

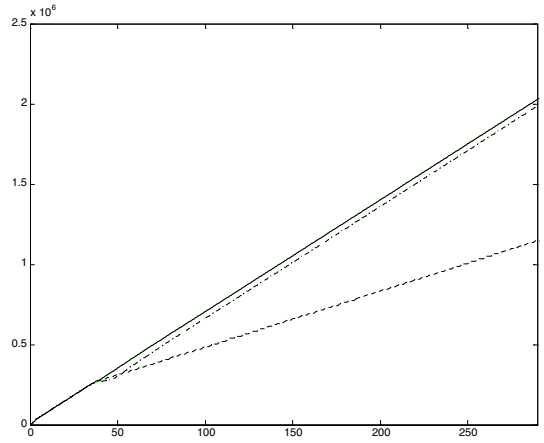

Fig. 3. TCP sequence number over wireless link. Solid line: Sack; dotted line: New Reno; dashed-dotted: Tahoe; dashed line: Reno. Sack and new reno are overlapped

and periodically introducing a polling bit in a PDU to request a status report about losses.

The TCP traffic fed to the input node in Fig. 2 is derived from measurements performed at the border gateway of the campus network of the University of Pavia [23]. It represents a web session of a generic connection randomly selected filtering the aggregate traffic stream.

For the results presented here, the UMTS error sequence used to generate the model is characterized by a E_b/N_0 = 3 dB, which corresponds to a BER of $3.44 \cdot 10^{-5}$: as said, in this case, a 2 states HMM suffices in generating the error process. When HMM generate an error event, the simulator is informed about the length of the incorrect transmission: the PDU transmission interval is compared with the error interval, obtained directly by the HMM: if the first is smaller than the error interval, then the PDU is not received correctly, while if larger, the transmission can be considered error free. In this last case, the HMM generates a new error event and so on.

The node B is informed of errors by a cumulative ack packet: in each one the transmission outcome of 5 PDUs is indicated. This simulates the source polling for error request. The polling PDUs size has been selected in order to reach a compromise between retransmission efficiency and network overload. Moreover, to avoid a TCP timeout, the cumulative ack size should be smaller than window transmission one. In case of error, the UMTS gateway will send the wrong PDUs in the next transmission window in the same positions, while the free slots will be used for transmitting next TCP packet.

The simulation time is about 300 s, which corresponds to about 2 millions of bytes: the length of the simulations allowed us to evaluate the different behavior of the TCP versions considered.

In Fig. 3, TCP sequence number over the wireless link is shown. These curves represent the amount of data transmitted by the source. As expected, the new reno and the Sack versions show good performances. Although they are very similar, the best result is shown by Sack, probably because it is able to better manage non consecutive errors..

Tahoe has a good performance too, even if the curve is lower than the other two versions. Finally, Reno has the poorest result, because of its attempt to recover from transmission errors, but its strategy is defective in case of error burst (multiple errors in the same window). After 300 s of simulations, while Sack, New Reno and Tahoe have sent about 2 millions of bytes, Reno has been able to send only the half.

In Fig. 4 a detail of the congestion window is shown: it represents the first 125 s of simulation, but it is able to show the different behavior of the four TCP versions. In the first 38 s they have a similar performance because of scattered

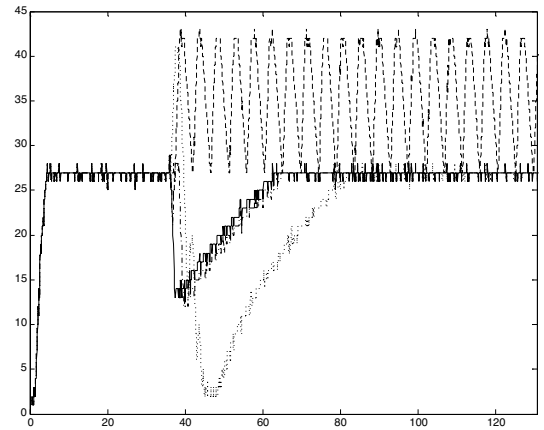

Fig. 4. Congestion window. Legend as in Fig. 4

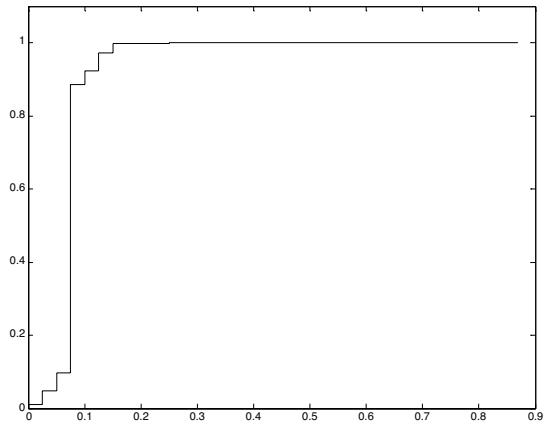

Fig. 5. Cumulative distribution function of ack interarrival times: note that all four curves are overlapped.

errors and absence of timeout events; after that a three duplicate ack event is generated, and different solutions are shown. Tahoe behaves as usual, putting the congestion window equal to 1, then increase it firstly in slow start (exponentially), and finally switches to congestion avoidance after reaching its threshold. New Reno and Sack succeed in avoiding slow start phase and, after halving their threshold, adopt congestion avoidance algorithm (with linear increments). The worse performance, again, is shown by reno: the agent, at the receiving of 3rd dup-ack, switches to fast recovery and increases congestion window by one each further dup-ack reaching the sender. Window exceeds the limit of 30 packets set in the simulation parameters due to the presence of previous packets stored in the buffer. This is not a real problem because the transimssion window is chosen as the minimum between congestion and advertised window. TCP Reno supposes that only a packet is lost, so, when retransmission of the datagram is confirmed, congestion avoidance algorithm is used. In case of a second corrupted packet, retransmission cannot be triggered immediately, it is necessary to wait for the timer expiration. During this period source stops sending data because of absence of acks (see "self clocking" mechanism), moreover time out forces the use of slow start algorithm. Tahoe can better manage multiple errors, even if less efficiently than evoluted versions of TCP: it always reacts to missing data in flow using slow start, so time out is no more needed for retransmitting packets. Congestion window (figure 4) remains 1 for the time necessary to serve pachets arrived before the duplicate one, as seen for Reno.

In Fig. 5, the cumulative distribution functions of the ack interarrival times are shown. Note that all four TCP versions have a similar behavior. This could lead to suppose that all have the same performance, but consider that the duplicate ack are included too. All ack packets are close each other, in all simulations and, considering that the return path is lossless and error free, this behavior should not be surprising. [24]

CONCLUSIONS

In this work the performances of TCP in a wired-cum-wireless scenario have been analyzed. Four version of TCP have been considered, and their behavior in a UMTS UTRA FDD environment has been evaluated, using the error model used is a 2 state HMM. The results show that the better performances are obtained by new reno and selective ack, which take into account of the error bursts over the wireless channel. Also tahoe shows good results, probably because of its simplicity, while reno is the worst version seen. In this last case, in fact, the tricks used to overcome the error bursts are ineffective, and even harmful.

Further investigation is needed to analyze performances in presence of more than one user, and also to study newer TCP version behaviors and toward the TDD options of the UTRA interface.

REFERENCES

[1] http://www.3GPP.org

[2] http://www.umts-forum.org

[3] B. Walke, *Mobile Radio Networks*, Second Edition, J. Wiley & Sons Ltd., Chichester, GB.

[4] H. Holma, A. Toskala, *WCDMA for UMTS: radio access for third generation mobile communicatios*, J. Wiley and Sons Ltd. Chichester, GB.

[5] Y.-B. Lin, A.-C- Pang, Y.-R. Haung, I. Chlamtac, "An all-IP approach for UMTS third generation mobile networks," IEEE Network, Sept./Oct. 2002, pp. 8-19.

[6] A. Chockalingam, M. Zorzi, V. Tralli, " Wireless TCP performance with Link Layer FEC/ARQ," *IEEE ICC 2001*, St. Petersburg, 11-15 June 2001.

[7] A.-F. Canton, T. Chahed, " End-to-end reliability in UMTS: TCP over ARQ," IEEE Globecom 2001, San Antonio, TX, USA, 25-29 Nov, 2001, vol 6 pp. 3473-3477.

[8] S. Heier, D. Heindrichs, A. Kemper, "Performance evaluation of Internet applications over the UMTS radio interface," *IEEE 55th Vehicular Technology Conference*, 2002, Volume 4 , 2002, pp. 1834-1838.

[9] K. Pentikousis, "TCP in wired-cum-wireless environment," *IEEE Comm. Surveys*, http://www.comsoc.org/pubs/surveys, Fourth Quarter 2000.

[10] 3GPP TS 25.212, "Multiplexing and channel coding (FDD)".

[11] 3GPP TS 25.322, "RLC protocol specification".

[12] Zhang Q., Su H-J, "Performance of UMTSRadio Link Control," *Proc. Of IEEE ICC2002*, New York City, 28 Apr.-2 May, Vol. 5.

[13] http://www.isi.edu/nsnam

[14] V. Jacobson, "Congestion avoidance and control," *Proceedings of ACM SIGCOMM '88*, Aug. 1988, Stanford, CA, pp. 314-329.

[15] RFC 2581: "TCP congestion control," Apr. 1999.

[16] RFC 2018: "TCP selective acknowledgements options," Oct. 1996.

[17] RFC 2582: "The New Reno modification to TCP's fast recovery," Apr. 1999.

[18] E.N. Gilbert: "Capacity of a burst-noise channel.," The Bell System Tech. J., vol 39, pp. 1253-1256, Sept. 1960.

[19] L.A. Grieco, S. Mascolo, "Performance evaluation and comparison of westwood+, new reno, vegas TCP congestion control," *ACM Sigcomm Computer Communications Review*, Vol. 34, No. 2, Apr. 2004, pp. 25-38.

[20] E. Costamagna, L. Favalli, P. Savazzi, F. Tarantola, "Performance analysis of TCP traffic over UTRA-FDD channels," *Proc. of WPMC 2003*, Yokosuka JP, 19-22 Oct. 2003.

[21] L.R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition", Proceedings of the IEEE, Volume: 77 Issue: 2, Feb 1989, Page(s): 257 –286

[22] Umbert A., Diaz P., "A radio channel emulator for WCDMA, based on the Hidden Markov Model (HMM)," *Proc. of IEEE VTC Fall 2000*, 24-28 Sept. 2000, Boston, pp. 2174-2179.

[23] E. Costamagna, L. Favalli, F. Tarantola, "Modeling and analysis of aggregate and single stream Internet traffic," *Globecom 2003*, San Francisco, CA, 1-5 Dec. 2003.

[24] Chadi Barakat, Eitan Altman, and Walid Dabbous, "On TCP Performance in a Heterogeneous Network: A Survey", IEEE Communication Magazine, January 2000, pag 40 – 46.