IMPROVED HEADER COMPRESSION FOR TCP/IP OVER WIRELESS LINKS

A. Giovanardi, G. Mazzini, M. Rossi, M. Zorzi

This letter investigates the TCP/IP Header Compression (HC) problem on wireless channels. After a brief review of the existing HC algorithms, two novel HC schemes are proposed, discussed and tested by means of simulations. The proposed schemes outperform the well-known approaches for all possible channel conditions and result to be very closed to the ideal case of HC without loss of synchronization.

Introduction: Header Compression (HC) is a technique to reduce the header overhead on band limited links. It is a well-known technique for point-to-point wired links, especially when the IP version 6 is considered, and it is becoming popular in IP based GPRS and UMTS cellular systems. This letter investigates the application of HC to the TCP/IP stack. The TCP/IP HC algorithm implemented in most TCP/IP hosts is the one proposed by Van Jacobson (VJHC) in [1], due to its low header overhead (3 to 5 bytes). VJHC works well on wired systems, e.g., serial lines, which are characterized by low packet error probability. In VJHC the HC is obtained by sending the difference between the current and the previous header (header delta fields) and by updating the reference value (header base) for each packet sent. Further studies have shown that the advantage of VJHC decreases when a wireless environments is considered [2] [3] [4], due to high packet error probability which leads to frequent HC synchronization loss, i.e., the event in which the reference header base used to compress is not available at the receiver.
A first algorithm to improve the HC synchronization robustness has been proposed by Degemark (DEHC) [2], who suggests a strategy to recover from the loss of synchronization between the compressor (C) and the decompressor (D). When synchronism is lost D assumes that this is due to the loss of a single packet $(n-1)$ and that its header delta fields are equal to those of the following one (n). The correct base is then computed by using this prediction, i.e., by adding twice the header delta fields of packet n to the header base [1], while ignoring the contents of packet $n-1$. If this attempt fails, a resynchronization request is sent by using ACKs flowing from D to C. DEHC needs considerable computational effort.
A second algorithm which aims at reducing the probability of synchronization loss has been presented by Perkins (PEHC) [3], where the header base is updated less frequently (every $K > 1$ packets) with respect to VJHC (for which $K = 1$), and each update is performed by sending the uncompressed header base. Thus, unlike in VJHC, in the PEHC algorithm the loss of a packet does not necessarily lead to loss of synchronization. K is set as large as possible, while respecting the maximum size of the binary representation of the header delta fields [1].
Calveras has presented a third algorithm (CAHC) [4], optimizing PEHC by means of an adaptive choice of K as a function of the channel behavior, i.e., K increases as the channel error rate grows.

Proposed Algorithms: In PEHC and CAHC, when an uncompressed packet or a new header base are lost the synchronism is lost and the system performance degenerates to VJHC.
In such a situation, in order to improve the performance we propose to continue using the current header base until the new one is successfully updated at the receiver. When C decides to update the header base, it sends the new base to D as difference with respect to the actual one and sets a suitable flag, called Request Flag (RF). By considering the acknowledgment nature of the TCP, when D receives the new header base, an ACK is transmitted from D to C. Only upon receiving this ACK can C start using the new base for compression. Let us note that this information does not introduce any overhead in the TCP signalling, since RF can be accommodated in one of the unused flags currently available. The next packet transmitted by C is encoded by using the new header base and this change is notified to D by means of a Bistable Flag (BF) that implements a sliding window mechanism of length two on the header base change.
By observing that RF and BF transitions occur at different times, it is possible to encode them into a single flag by using the exclusive-OR operator. This resulting flag is sent from C to D by means of the unused flag of [1]. The header base update is performed as frequently as possible to minimize the delta fields resource occupancy. Since the header base update involves two adjacent ACKs, this is also assumed as the rate of the header base change. We remark that in our proposal K is implicitly set as a function of the channel conditions. We also stress the fact that the BF and ACKs joint mechanism, unlike the other proposed schemes, guarantees that synchronization losses never occur.
As a consequence of an error, TCP retransmits all unacknowledged packets and C does not compress their header. The HC mechanism restarts with the first new packet (after the last retransmission). This novel algorithm, implementing the above described header base mechanism and retransmission strategy, is called Adaptive Base Proposal with Static Compression Start (ABP-SCS). Another possible scheme that we propose continues to transmit the uncompressed packets until C receives the ACK relative to the last retransmitted packet, to avoid the performance degradation due to HC in error prone channels. By doing so, some correct packets may also be transmitted with uncompressed headers. We call this strategy Adaptive Base Proposal with Dynamic Compression Start (ABP-DCS).

Simulation Results and Conclusions: The reviewed and the two proposed HC algorithms are compared by means of simulation. VJHC will not be reported due to its well-known bad performance on wireless channel.
The wireless channel is modeled by using a discrete time two-state Markov chain [5] where a slot time covers a single packet transmission. The characteristic parameters are the packet error probability, P_{ep}, and the average number of consecutive packet errors, b.
The following parameters have been considered: channel data rate of 144 Kbps, fixed payload length of 512 bytes, single path propagation delay of 16 ms, TCP over IP version 4 stack with a total of 40 bytes for an uncompressed header.
A unidirectional link is considered, i.e the packets flow from C to D, whereas in the reverse direction only the ACKs flow. The direct link packet generation is assumed continuous, as in a long FTP session. We use the New Reno algorithm. To simulate CAHC, we have considered PEHC by selecting the K value which maximizes the throughput within the range from 1 to K_{max}, where K_{max} is defined as in [3]. Furthermore, we report the TCP/IP New Reno performance without HC. In the case labeled IDEAL, VJHC is considered under the assumption that packet losses due to channel errors affect the TCP operation but never lead to loss of synchronization. This case is obviously unrealistic but is expected to provide an optimistic bound to all other schemes, as discussed in [2], in view of the fact that VJHC is the HC scheme providing the best compression performance in TCP/IP wired environment.
Figures 1 and 2 report the average throughput (defined as the fraction of the raw channel transmission rate which provides correct information bits at the receiver, i.e., not counting erroneous transmissions, spurious retransmissions, idle time and overhead) as a function of b with $P_{ep} = 0.001, 0.177$, respectively. As expected, in all cases the IDEAL scheme shows the best performance. The performance of all algorithms degrades as P_{ep} increases but TCP/IP exhibits a more robust behavior due to the complete absence of the synchronization loss problem (no HC is used in this case). In general, for a given value of P_{ep}, the throughput increases as the average number of consecutive packet errors increases, due to the decreased frequency of error events. The proposed ABP-SCS and ABP-DCS schemes give always better performance than the other HC algorithms and close to the selected IDEAL case.
The two proposed algorithms have performance quite close to each other except for bad channel condition, where at low b values ABP-DCS works better than ABP-SCS, thanks to the

ability to stop HC. However, a set of system parameters exists where ABP-SCS outperforms ABP-DCS. This trend is due to the trade-off between higher overhead and synchronization robustness of the ABP-DCS scheme.

Conclusion: Two different TCP/IP HC algorithms for wireless links have been proposed and tested by means of simulations, showing better performance than all known HC schemes. Even if the throughput gain with respect to the CAHC algorithm (considered the best) is little, the gap to reach the IDEAL performance has been significantly reduced in many channel conditions, which results in higher performance, especially in long FTP sessions.

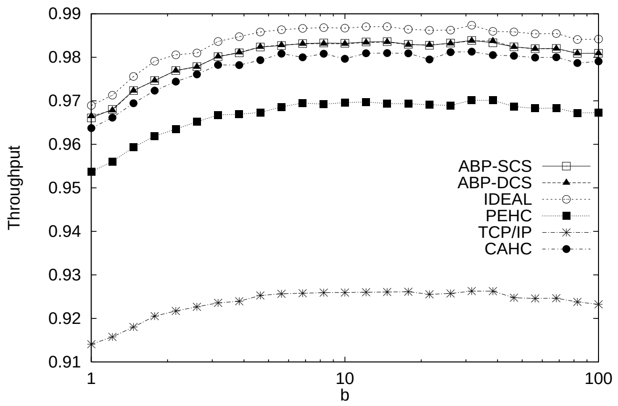

Figure 1: Throughput as a function of b for $P_{ep} = 0.001$

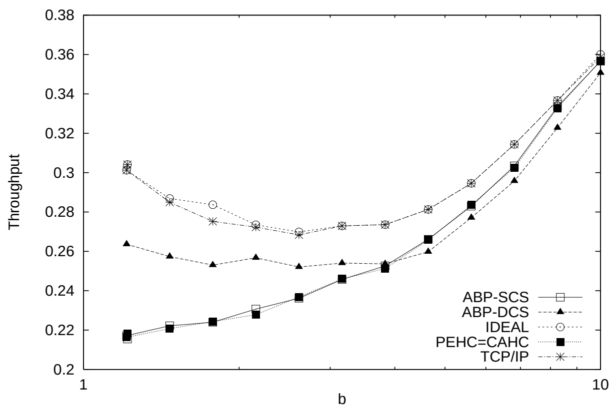

Figure 2: Throughput as a function of b for $P_{ep} = 0.177$

References

[1] Van Jacobson, “Compressing TCP/IP headers for Low Speed Serial Links,” IETF RFC 1144, Feb. 1990.

[2] M. Degemark, M. Egan, B. Nordgen, S. Pink, “Low loss TCP/IP Header Compression for Wireless Networks,” in proc. of ACM Mobicom96.

[3] S.J. Perkins and M.W. Mutka, “Dependency Removal for Transport Protocol Header Compression over Noisy Channels,” in proc. of ICC’97.

[4] A. Calveras-Aug, M. Arnau-Osorio, J. Paradells-Aspas, “An Improvement of TCP/IP Header Compression Algorithm for Wireless Links”, in proc. of ISAS’99.

[5] M. Zorzi, R.R. Rao, L.B. Milstein, “Error statistics in data transmission over fading channels,” *IEEE Trans. Comm.*, vol. 46, pp. 1468-1477, Nov. 1998.