

# Cooperative IP Header Compression for Parallel Channels in Wireless Meshed Networks

Frank H.P. Fitzek<sup>†</sup>, Tatiana Kozlova Madsen<sup>†</sup>, Petar Popovski<sup>†</sup>, Ramjee Prasad<sup>†</sup>, Marcos Katz<sup>‡</sup>

<sup>†</sup> Department of Communications Technology, Aalborg University

Neils Jernes Vej 12, 9220 Aalborg Øst, Denmark,

phone: +45 9635 8678, e-mail: [fff—tatiana—petarp—prasad]@kom.auc.dk

<sup>‡</sup> SAMSUNG Korea

**Abstract**—In this paper we introduce a novel header compression technique for parallel channels as they are found in meshed networks. The approach introduced advocates the cooperative behavior of parallel channels to maintain the compression state on sender and receiver side. The general concept as well as one particular implementation are shown. Our approach is characterized by no need of a feedback channel and a low complexity of a compression strategy. By means of analytical study combined with our testbed results we show the performance of the introduced approach. The designing goals were to have a low complex, highly efficient and also robust header compression dealing with the characteristics of the wireless channel. As a first result we can show that for independent error pattern a number of three cooperative channels achieves both robustness and efficiency for a wide range of errors.

## I. MOTIVATION

Currently research is focusing on the exploitation of multiple channels in meshed networks for ongoing communication between two entities. Compared with only one channel, multiple channels allow more flexibility, robustness, and higher capacity. Here and hereafter a channel is the physical description of a resource that is used for communication. An examples for multiple channel communication can be found in wired and wireless multi hop networks. Figure 1 shows a meshed network example with three channels. Each channel (such as channel one) is composed of multiple hops ( $C_1 C'_1 C''_1 C'''_1$ ) between sender and receiver. The channels are characterized by delay, bandwidth and error rate.

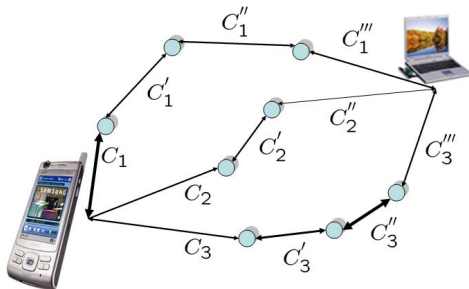


Fig. 1. Meshed network example.

Novel applications support multi channel communication. In a video conference scenario, the audio and the video stream are transported over different RTP/UDP/IP streams. Further

separation of the streams is on the way. Multiple description coding (MDC) or Multi Layered Coding (MLC) have gained a lot of interest lately. MDC allows splitting a single source (e.g. audio or video) into multiple descriptors, where each of the descriptors may be send over different channels. At the receiver each descriptor is decodable in a stand-alone fashion. The more descriptors are received on receiver side, the better the quality becomes. MLC generates also multiple entities of an information source, where two types of entities are distinguished. The *base layer* of MLC can be decoded in a stand-alone fashion, while the *enhanced layers* contribute to a better quality. Without the base layer, the enhanced layers are useless. Both, MDC and MLC, have their advantages and disadvantages and it depends on the scenario which one fits better. Nevertheless the trend is obvious that multiple streams are getting more importance in future communication networks.

For the transportation of the descriptors or layers over IP networks the related overhead in terms of additional header information has to be taken into account. It depends on the scenario, whether it is real-time or streaming, how large the IP datagrams are. Nevertheless for each datagram a full IP header (and its related protocols) has to be taken into account. Due to the large overhead of IP communication header compression becomes very important. For multimedia services the overhead using the Real Time Protocol (RTP) [1], the User Datagram Protocol (UDP), and the Internet Protocol (IP) itself is 40 byte for the IP version 4 and 60 byte for IP version 6.

In case for real time voice communication the overhead becomes dramatically large due to the small payload size. But even for video communication the overhead is still not negligible. In Figure 2 the application and network overhead<sup>1</sup> for the well known foreman video sequence using MDC is given. Three different quantization values (namely 1, 31, and 51) versus the number of parallel descriptors (sub-streams) is given for IP version 4. The interested reader is referred to [2]. Low quantization parameters refer to high quality video, while higher quantization values are reducing the video quality.

In Figure 2 the curve entitled QP 1, QP 31, and QP 51 gives the overhead that is introduced only by the encoding

<sup>1</sup>The overhead is defined as the ratio of amount of data for all sub-streams divided by the amount of data for a single stream.

process. While the QP  $X$  + Network gives the overhead of the encoding process with quantization level  $X$  in addition to the network overhead. It can be seen that the overhead increases dramatically due to the network overhead and not due to the encoder overhead if large quantization values are used. For small quantization values the impact is less. But for efficient video encoding the quantization values will be large (between 31 and 51) assuring both high video quality and low data rates.

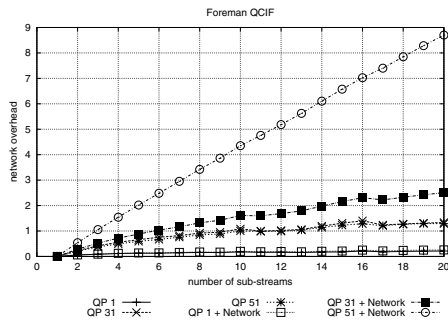


Fig. 2. Network Overhead (RTP/UDP/IPv4) for the foreman video sequence and three different quantization values.

Note, that for the quantization value 51 and 20 descriptors the overhead introduced by the encoder equals 1.3 (i.e. more than twice the bandwidth is used, see later section for overhead definition). If the network overhead of IPv4 is also taken under consideration, the overhead increases up to 8.8 (encoding plus network, thus the bandwidth is ten times higher in contrast to the single channel case). Therefore if we want to use the advantages of multiple channel communication such as MLC or MDC, header compression is crucial.

Our approach focuses on header compression on multiple channels. The designing goal of our approach is high bandwidth efficiency and robustness against packet loss. The parallel channels work in a cooperative way to support each other. In our first investigations we assume that all  $J$  channels between sender and receiver have the same delay characteristics (within the granularity of IP packets). Thus we refer to synchronous channels. In case IP datagrams are segmented into smaller data link packets for transmissions and conveyed to the receiver in a reliable fashion, channels errors lead to jitter between the IP packets. On the receiver side we need a buffer of some IP packets for each cooperative channel. The buffer is needed to explore the full advantage of our approach and in case that one application is using multiple channels, these channels are inherently synchronized.

Even if we have started our motivation with the MDC or MLC, we want to note that this approach can be applied in any case where multiple channels are used between sender and receiver. One example is the construction of a WEB site. Multiple Transport Control Protocol (TCP) connections are used (for text, pictures, etc). In the following we take MDC

as an exemplary service to show the benefit of our approach.

## II. HEADER COMPRESSION FOR WIRELESS COMMUNICATION

Before we present our concept, we want to introduce the main concept and terminology of header compression research. Header compression is possible due to some redundancy among the different header fields of different protocol layers and the interdependencies of IP packets. The general concept of header compression is given in Figure 3. On sender-side the original header is compressed by using the current sender base. The base is the knowledge of previous send packets. The compressed header is conveyed via the wireless channel to the receiver. On the receiver side the compressed header is decompressed by using the current receiver base. For a successful decompression both bases need to be the same.

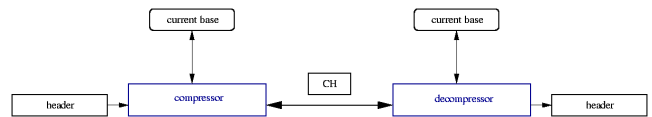


Fig. 3. General concept of header compression: A current base header is maintained at compressor and decompressor. Header redundancy with respect to the base header is removed to obtain the compressed header (CH).

One possibility of header compression is delta coding. This approach is often chosen (as by Van Jacobson in 1990 [3] RFC1144) because of its low complexity and still quite good performance. In this approach one uncompressed header is sent and followed by a row of compressed headers. The compressed headers are carrying only the differential information referring to the preceding header, where the differences between two packet headers are referred to as the *delta*.

For error-free channels this is a very efficient way of header compression. Only in case one header got corrupted the following compressed headers are also error-prone. This is due to the corrupted base knowledge on receiver side. To introduce robustness and therefore heal the base on receiver side, uncompressed headers are transmitted to update the receiver base again. The price of robustness is the efficiency of compression.

A further approach for robust transmission is specified in RFC3095 [4]. The approach is called Robust Header Compression (ROHC). The drawback of this approach is the high memory usage and the need for a feedback channel in two out of three modes. Another header compression is specified in [5]. But all header compression schemes including [3-5] focus on single links only.

## III. COOPERATIVE HEADER COMPRESSION

### A. State of the Art

In Section II we have already introduced existing header compression schemes. Here we want to show the application of existing header compression solutions to multiple channel communication. In Figure 4 the separate delta coding as introduced in [3] is applied to parallel channels. In this approach

each header compression entities act independently from each other. Transmission is done by frames of  $N$  packets, where only the first packet is sent with an uncompressed header. This approach simply combines the usage of parallel channels with delta coding.

In case channel errors occur the packets are lost. Furthermore the current base is not updated properly and all consecutive compressed headers will be error prone even if the packet itself was received successfully (as for the single channel case). To show this effect we apply dashed crosses (error prone packets due to error propagation) and solid crosses (lost packets due to channel condition) in Figure 4. In both cases the payload is lost. To prevent unlimited loss propagation, uncompressed header is sent periodically to refresh the base.

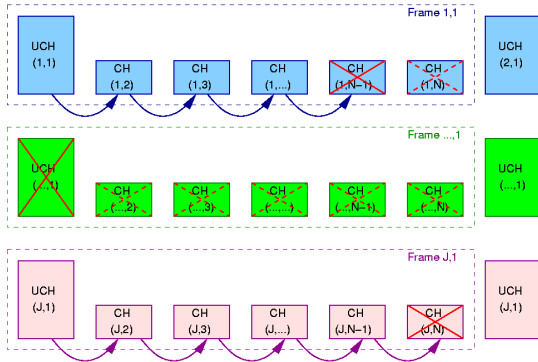


Fig. 4. Separate Delta Coding (DC) on each parallel channel with packet errors (three times) and the related error propagation (six times).

### B. The Proposed Header Compression Scheme for Multiple Channels

We propose a scheme where each header compression entity is sending a header for its ongoing compression and furthermore carry some additional information for the neighboring channels. We introduce the concept of an additional information container (AIC). The AIC is used to repair a corrupted current base of neighboring compression entities. It is not the purpose of the AIC to repair the whole packet (including the payload) but to retrieve the current base.

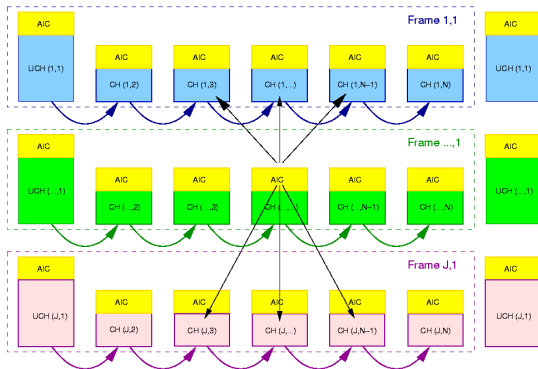


Fig. 5. General approach of piggy-backed information for parallel channels.

In Figure 5 the transmission of the AICs is given. Additionally we show how one AIC can carry information for each header. There is no limitation in the channel or time domain. By introducing AICs we reduce the problem of error propagation and therefore we may reduce the refreshing rate (increasing  $N$ ). For efficient coding the design of the AICs is very important. In other words we would like to reduce the overhead and simultaneously minimize the packet error probability. There is a trade-off between these two parameters.

It is possible to rebuild a current base (even if the current packet is lost due to channel conditions) by using the AIC information in combination with any given base. The AIC for a given channel does not necessarily refer to the base of this given channel. This is up to the designing process. Obviously, the size of the AIC is smaller if it refers to the base of the channel to which it is dedicated to support. One possible solution would be that the AIC is just a copy of a compressed header and send over different channels. In this case the AICs are send on parallel channels and refer to the base of the current channel. We note that this approach is less complex, but not the most efficient one. To make it more efficient, the amount of data has to be reduced. In comparison to a compressed header the AIC do not have to carry any information that belongs to the packet itself, but only everything that is needed to retrieve the base. An example is the UDP checksum, which should not be part of the AIC. The UDP checksum is needed to check whether the packet is received correctly. This would decrease the amount of data for each AIC.

In Figure 6 one possible implementation of piggy-backed information (AIC) for neighboring channels is given. In addition to the normal header for each packet, the header compression entity includes one AIC for each neighboring channel in the same time domain. In this approach only AICs with the same time instants can be used to repair the base in case of packet errors. This helps to reduce the amount of data for the AICs. Note, that this seems to be very efficient as long we have synchronous channels. In case of asynchronous channels we should consider the use of time-domain separated AICs.

In case of packet errors due to channel conditions, the probability of error propagation is reduced as given in Figure 6. The example shows that the two packet errors on the first channel caused by channel conditions, has no impact on the following compressed headers. Even if the whole packet (1,3) on channel 1 is lost, we do not apply a cross to the compressed header  $CH(1,3)$  as it can be healed by  $AIC(1,3)$ . The base can always be repaired using the AICs carried by the neighboring channels. In this example we have five packet errors due to propagation errors, but no packet lost due to error propagation.

Note, there is still the possibility that headers got corrupted due to error propagation if and only if all  $J$  channels are error prone at the same time. Obviously this probability decrease with larger number of  $J$ . Highest bandwidth efficiency can be retrieved if the parameter  $N$  is chosen in dependency of the

channel error probability (CEP).

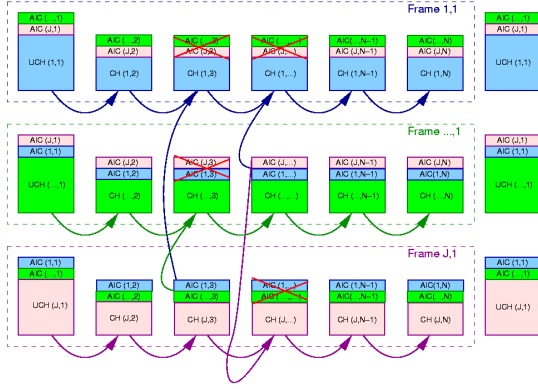


Fig. 6. One possible implementation of piggy-backed information (AIC) for neighboring channels with packet errors.

#### IV. THEORETICAL ANALYSIS

This section presents the analytical evaluation of the different strategies for header compression on cooperative channels. For our first results we assume independent error patterns on the synchronous channels. A robust scheme is the one with low Packet Error Probability (PEP). Here we would like to underline that using different differential coding scheme a packet can be lost either because of the error due to bad channel condition or because a receiver does not have a base to decompress the header and therefore a packet is of no use. We will clearly distinguish these two causes of errors in our analysis.

The price to pay for decreasing PEP is the increase in overhead. It is shown that even though the proposed algorithms require additional overhead, they can achieve much higher bandwidth efficiency compare with unmodified delta coding scheme. If for a particular case it is important to have a very robust header compression, AICs can be appended to packets on each of the synchronous channels. In this case, when number of channels is sufficiently large, probability to lose a packet due to the absence of a base is almost zero. In other words, a packet is lost only when a channel is bad - this situation is unavoidable even if no header compression is used.

##### A. Analysis of Delta Coding

In the delta coding, a loss of one packet results in the loss of all subsequent packets in the frame as given in Figure 4. In other words, if  $k$  packets from a  $N$ -packet frame are lost, it means that the first  $N - k$  packets are received correctly; a packet number  $k$  has not been received (due to the channel error); and the last  $k - 1$  packets could not be decompressed. Therefore, PEP can be found as

$$PEP = 1 - \frac{(1-p)(1-(1-p)^N)}{Np} \quad (1)$$

We define the bandwidth efficiency  $BE$  as the amount of successfully received useful information over the total amount of transmitted information:

$$BE = \frac{D(1-p)(1-(1-p)^N)}{p(ND + B_u + B_c(N-1))}$$

where  $D$  is payload in bytes (we assume all packets have the same payload size),  $B_u$  and  $B_c$  are size of uncompressed and compressed header respectively.

##### B. Analysis of Cooperative Header Compression

Now we consider the cooperative scheme when each of the packet on  $J$  synchronous channels carries additional information (AICs) that helps to reconstruct the base. In contrast with a delta coding, the only situation that can entail error propagation is the loss of packets on all channels. Taking into account that a packet can be both corrupted or undecodable, the following formula for PEP can be obtained:

$$PEP = 1 - \frac{(1-p)(1-(1-p^J)^N)}{Np^J} \quad (2)$$

One can notice that substituting  $J = 1$  (3) we get formula (2). It is also worth noting that when  $J \rightarrow \infty$ ,  $PEP \rightarrow p$ . This confirms our intuition: increasing number of channels, we decrease probability of errors. Using robust modification of delta coding we can eliminate propagating errors, but we can not combat channel errors, thus, probability (3) asymptotically approach the probability to lose a packet due to the channel error.

Expression for the bandwidth efficiency is given by

$$BE = \frac{ND(1-PEP)}{(ND + B_u + B_c(N-1) + N(J-1)B_a)}$$

where  $B_a$  is a size of one AIC.

#### V. PERFORMANCE EVALUATION

##### A. Packet Error Probability

To test our analytical results, a testbed has been developed. The testbed consists out of two PCs where IP traffic is conveyed from sender to receiver with the possibility of cooperative header compression. The channel errors are emulated at the sender side, while the performance in terms of PEP is measured at the receiver side. Due to space limitation we refer only in this subsection to our testbed. Further details will be presented in future work.

In Figure 7 dependence between PEP and CEP is shown. Theoretical results, presented by curves, and obtained measurement results, presented by points, are in very good correspondence with each other. When no compression scheme is used, then PEP is equal to CEP. The line  $PEP = CEP$  gives us a lower bound on the error probability. From Figure 7 one can observe that by increasing number of cooperative channels  $J$  we approach the lower bound and already for  $J = 3$  the corresponding curve lies very close to the uncompressed case curve.

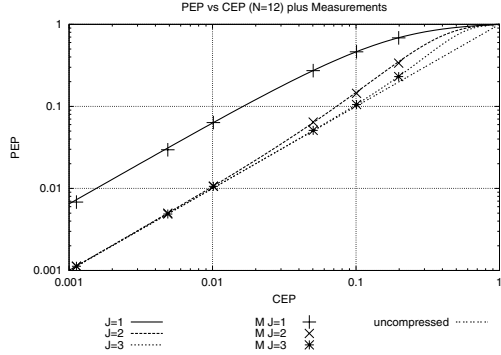


Fig. 7. Analytical and measurement results for PEP versus CEP with  $N=12$ .

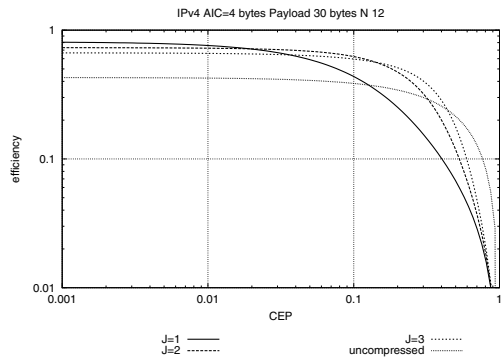


Fig. 8. Efficiency for small payload sizes (30 bytes) versus CEP for  $N=12$ .

### B. Efficiency

Figure 8 shows efficiency as a function of a channel error probability for different number of cooperative channels. Under very good channel conditions, curves for different  $J$  lie close, but the simple delta-coding scheme ( $J = 1$ ) performs the best: in this case almost all packets are received correctly, thus using AICs we just introduce additional unnecessary overhead.

When CEP is larger than the threshold of approx. 2%, the efficiency of the delta-coding compression drops rapidly. For the large range of CEP the cooperative compression scheme with  $J = 2$  and  $J = 3$  has the best performance. Having channels errors more than 10%, we should use 3 channels to achieve the high efficiency. On the other hand, operating in the state when packets are corrupted due to the channel errors with the probability close to 1, the best choice is not to use header compression schemes at all (note that the efficiency of packet transmission is very low in this case).

## VI. CONCLUSION AND OUTLOOK

We have introduced a novel header compression scheme for multiple channels. Our approach relies on the cooperative

behavior of the parallel channels. The general concept of additional information containers is introduced. Using different AICs implementation strategies, a corrupted compression base of neighboring entities (either in channel or time) can be repaired. Even though by using AICs additional overhead is introduced, it was shown that the overall efficiency and robustness of the compression scheme is significantly improved. To summarize, the advantages of the presented approach are the following:

- high bandwidth efficiency
- low memory consumption (low cost terminal)
- low complexity (low cost terminal)
- robustness
- no need of feedback channel (low cost terminal)
- can be used with every overlaying protocol stack (e.g UDP and TCP)
- exploitation of multiple channels
- only a small number of cooperative channels are needed to perform efficiently

From the presented results, we conclude that the number of cooperative channels should be larger than two to achieve best robustness and efficiency in case independent packet errors are assumed. A total number of three cooperative channels seems to be sufficient. This does not mean we limit the number of parallel channels to any number. In case nine descriptors would be perfect to support the needed services, we would split them into three groups with three cooperative channels.

Currently, we are verifying the obtained theoretical results by simulations and measurements obtained with the help of the testbed. We are investigating the impact of the correlated channel errors on the performance of the proposed compression scheme. Both correlations in time and between channels will be considered. Within the further work scope there is a design of more sophisticated implementation methods and construction of AICs to achieve further performance improvement.

## VII. ACKNOWLEDGEMENT

This work has been supported by SAMSUNG, Korea. The authors would like to thank A. Rashid for his help with the testbed.

## REFERENCES

- [1] H. Schulzrinne, "RTP Profile for Audio and Video Conferences with Minimal Control", Request for Comments 1890, January 1996.
- [2] F. H. P. Fitzek, "Application and Network Measurements For Multiple Description Coded Video Sequences", Tech. Rep., Aalborg University, February 2004, ISBN 87-90834-46-1.
- [3] V. Jacobson, "Compressing TCP/IP Headers for Low-Speed Serial Links, Request for Comments 1144, February 1990.
- [4] C. Bormann, C. Burmeister, M. Degermark, H. Fukushima, H. Hannu, L-E. Jonsson, R. Hakenberg, T. Koren, K. Le, Z. Liu, A. Martensson, A. Miyazaki, K. Svanbro, T. Wiebke, T. Yoshimura, and H. Zheng, "Robust Header Compression: Framework and four profiles, Request for Comments 3095, July 2001.
- [5] S. Casner and V. Jacobson, "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links, Request for Comments 2508, February 1999.