Two Hybrid ARQ Algorithms for Reliable Multicast Communications in UMTS Networks

George D. Papadopoulos, *Student Member, IEEE*, Georgios Koltsidas, *Student Member, IEEE,* and Fotini-Niovi Pavlidou, *Senior Member, IEEE*

***Abstract*— Multicast communications in wireless networks have attracted the interest of the scientific community, since many issues remain open. In this framework, our letter proposes two novel Hybrid Automatic Repeat Request (HARQ) algorithms for multicast communications in the UMTS Radio Link Control (RLC) layer. The proposed algorithms exploit the channel autocorrelation in order to dynamically estimate the multicast users' channel conditions and thus reduce the mean Service Data Unit (SDU) delay and increase the SDU throughput.**

***Index Terms*— UMTS, multicast, Hybrid ARQ, packet FEC.**

I. Introduction

The development of Universal Mobile Telecommunications System (UMTS) promises high communications bandwidth to end users, while supporting a great variety of new services. A popular subclass of these new services requires multicast communications, instead of the existing unicast ones. However, multicasting raises the issue of efficient exploitation of network resources. In this framework, the 3rd Generation Partnership Project (3GPP) has been working on an overall multicast scheme that covers almost all network layers, called Multimedia Broadcast/Multicast Service (MBMS) [1].

The UMTS Radio Link Control (RLC) layer exists in the Radio Network Controller (RNC) and in the user equipment (UE). The RLC, as a link layer mechanism, aims at ensuring the reliable transmission and reception of packets between the RNC and the UE. Since the wireless medium introduces a significant amount of errors, error control techniques have been used to ensure the correct reception of packets.

In this letter, two Hybrid Automatic Repeat Request (HARQ) algorithms are proposed for multicast communications in the UMTS RLC layer. The proposed algorithms exploit the channel autocorrelation in order to dynamically estimate the multicast users' channel conditions and thus reduce the mean Service Data Unit (SDU) delay and increase the SDU throughput.

II. Packet Level FEC

In point-to-multipoint communication networks, the Forward Error Correction (FEC) technique is desirable as a complement to ARQ [2]. In [3] Reed-Solomon erasure correction code is examined at the packet level, while in [4] and [5], the concept of packet level FEC is described. Using the same philosophy as in traditional FEC, packet level FEC handles packets as if they were bits. To better explain the procedure, let us assume that K information packets are used to produce H redundant packets. If we denote as $X_{1..K}$ the information packets and $Y_{1..H}$ the redundant packets, then from [4]

$$Y_j = \underset{i=1..K}{\oplus} \left(X_i \ll (i^{j-1} - 1) \right) [1]. \quad (1)$$

The $N = K+H$ packets (which form a Transmission Group – TG) are transmitted over the Common Channel (CCH). At the receiver, the TG can be decoded, if at least any K out of the N packets are received correctly. Under HARQ, besides the packet level FEC which is used for error correction, a cyclic redundancy check (CRC) code is calculated for every packet and included into the packet field, in order to detect errors [5].

III. Hybrid ARQ Algorithms

In this section, three HARQ algorithms for multicast streaming, proposed in [6] (also investigated in [7]), are reviewed. Besides, two novel HARQ algorithms are presented that can be used at the RLC layer to improve both throughput and delay performance with respect to the plain ARQ scheme and to the three aforementioned algorithms.

A. Algorithm A1

At the sender's side, groups of K PDUs (Protocol Data Units) are encoded to $N = K + H$ PDUs, where H are the redundant PDUs, as explained in section II. The N PDUs are sent to the multicast users over the CCH. The K data packets can be recovered by each multicast user if and only if the number of erroneous PDUs received in a TG is less than or equal to H. In the case that at least one receiver is not able to correctly decode the N PDUs, he sends a negative-acknowledgement (NACK) message back to the transmitter, including the identifier for the TG that he could not decode correctly. Then, the sender retransmits the entire TG, until all the receivers are able to recover the K original PDUs.

B. Algorithm A2

N PDUs are sent to the multicast users through the CCH. Each user detects if he can correctly decode the TG, i.e. if the received erroneous packets are less than or equal to H, and if not, he sends a NACK message back to the transmitter with the

The authors are with the Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Panepistimioupolis, 54124 Thessaloniki, Greece. E-mails: {papgeo, fractgkb, niovi}@auth.gr.

This research was supported by the B-BONE (Broadcasting and multicasting over enhanced UMTS mobile broadband networks) European project.

[1] $\oplus$ indicates the "*exclusive or*" operation and $\ll$ indicates the "*left-shift*" operation.

TG identifier. If at least one receiver requires retransmission, the sender uses the K original packets to generate one new redundant PDU (incremental redundancy), which differs from all the previous ones, and sends it to all the multicast users through the CCH channel again. The receivers can now decode $N + 1$ PDUs if and only if K of them are correct. The entire procedure continues until all the receivers can correctly recover the K original PDUs, by decoding the $N+R_{tot}$ PDUs sent, where R_{tot} is the total number of retransmissions.

C. Algorithm A3

As in the previous schemes, N PDUs are sent to the multicast users at the beginning. If at least one user cannot recover the K original PDUs, he sends a NACK message back to the transmitter with the TG identifier. Additionally, each user $i \in \mathcal{N}$ (with $\mathcal{N}$ being the set of multicast users in a cell) sends back to the transmitter the minimum number R_i of the incremental redundant PDUs needed for the decoding of the N PDUs, i.e. $R_i = K - N_{ok}(i)$, where $N_{ok}(i)$ is the number of the correctly received PDUs for user i. The transmitter collects all the received R_i's and retransmits $R = max_{i\in\mathcal{N}}\{R_i\}$ redundant PDUs. If $N_{ok}(i) \geq K \ \forall i \in \mathcal{N}$, then the K PDUs can be recovered by the entire multicast group. Otherwise, if $N_{ok}(i) < K$ the algorithm continues until all the receivers can correctly obtain the K original PDUs.

D. Proposed Algorithm A4

Here we propose an algorithm that improves the performance of A2. The first K original PDUs are handled as in A2. Then, the next TG is sent to the multicast users. If $\min_{i\in\mathcal{N}}\{N_{ok}(i)\} \geq K$ the original K PDUs can be obtained and the TG is correctly received. Otherwise, the transmitter sends R_{tot} incremental redundant PDUs generated from the K original PDUs, where R_{tot} is the total number of the needed incremental redundant PDUs of the last TG transmission that needed a retransmission. If all the receivers recover the K original packets by decoding the $N + R_{tot}$ PDUs, R_{tot} is decreased. Otherwise, the transmitter generates one more new redundant PDU and transmits it to the multicast users, as in A2, until all the users obtain the K original PDUs. In the latter case, the R_{tot} becomes $R_{tot} = R_{tot} + R$, where R is the number of the additional retransmissions of one PDU each time, regarding this TG.

To better explain the algorithm, it should be mentioned that, regarding the first TG, the algorithm is identical to A2. For the following packets, it exploits the information from the previous TG, in order to make an estimation of the channel conditions, based on the fact that the channel does not change too fast and the expected overall error rate will be very close to the one experienced previously. Thus, the expected number of redundant PDUs is considered equal to the total number of redundant PDUs needed for the previous TG. Additionally, the algorithm adapts itself to the dynamics of the channel conditions and increases or decreases the number of retransmissions in the following TG transmissions according to the experienced conditions.

E. Proposed Algorithm A5

This algorithm is an improvement on A3 and its rational is very close to that of A4. The first TG is transmitted according to algorithm A3. For the next TG, though, the sender transmits $N' = K + R_{tot}$, instead of N, where R_{tot} is the total number of redundant PDUs required in the previous TG transmission. In this way, the algorithm dynamically adjusts to the system's requirements. There is also a mechanism that updates R_{tot}; if for a TG, the N' PDUs were enough and no further retransmission was required, the R_{tot} is decreased. In case that additional retransmissions were required, then R_{tot} is increased. The way R_{tot} is updated is of primary importance, since its performance is highly related to the mechanism of updating and can vary from minimum delay with low throughput to high delay with high throughput.

IV. Simulation Model

In our simulations we adopted the simple channel model used in [6]. A Discrete Time two state Markov Channel (DTMC) is used, where the two states correspond to the cases where the packet is correctly received, "*good state*", and the packet is erroneously received, "*bad state*". The transition from the "*good state*" to the "*bad state*" takes place with probability $p = 0.1$ and the channel remains in the "*bad state*" with the same probability p. According to this model, the probability that a packet is erroneous is p. We mention here that each user has its own channel, independent from all the channels of the rest of the users in the multicast group. The parameters of the packet level FEC are $K = 15$ and $N = 19$. We consider a RLC round trip time (RTT) of 80ms, a link layer logical bit rate of 240kbps, PDU length of 352 bits and SDU packet length of 500 bytes [6].

As concluded from section III, both A4 and A5 are based on variable parameters; the mechanism that decides on the number of additional PDUs is not fixed, rendering the algorithms more flexible. In our computer simulations we have chosen the following parameters, after optimizing the algorithms, in order to achieve a good trade-off between delay and throughput. In A4, the reduction of R_{tot} analyzed in section III-D is performed by the operation $R_{tot} = R_{tot} - 1$. In A5, R_{tot} is increased and decreased, as explained in section III-E, according to $R_{tot} = H + (r - R_{tot})/2$ and $R_{tot} = R_{tot} - 2$, respectively, where r is the total number of redundant PDUs transmitted in the previous TG. However, other mechanisms can stimulate different versions of both algorithms' performance.

V. Results and Discussion

In this work, two metrics were evaluated to assess the efficiency of the algorithms described in the previous section. The first one is the mean SDU delay, which is defined as the elapsed time from the transmission of the first PDU (belonging to the first TG of an SDU) until the reception of the last PDU related to the corresponding SDU, from all the users, with no further retransmissions needed, averaged over the total number of SDUs sent. The second one is throughput, which is considered as the number of original PDUs divided by the total number of PDUs sent. The plain ARQ mechanism is used

as a reference for the performance evaluation of the HARQ algorithms.

In Fig. 1 we present the mean SDU delay. A4 algorithm is related to A2, but it dynamically adjusts the required redundant PDUs based on the number of the redundant PDUs used in the previous TG transmissions. We notice that A4 outperforms A2, as well as A3 concerning this metric. A5 algorithm is based on A3, but it performs better in terms of average delay. Comparing A4 and A5, we observe that when the number of multicast users N_u is below 200, A5 outperforms A4. When $N_u \geq 200$, then A4 presents less delay than A5. This happens partially due to the update mechanism chosen for algorithm A5, which uses a bigger step when it increases and decreases the number of the redundant PDUs, while A4 uses a smaller step.

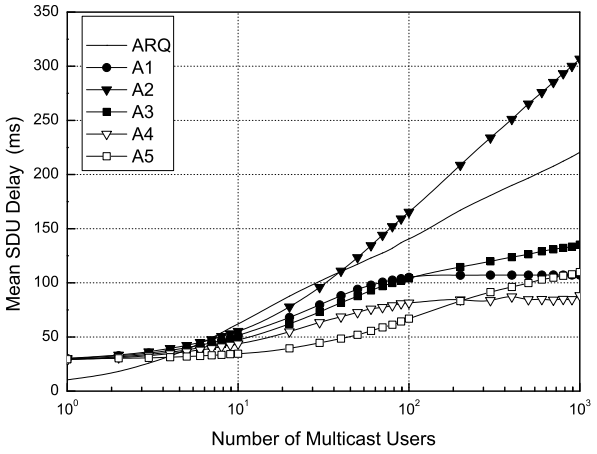

Fig. 1: Mean SDU delivery delay time as a function of the number of multicast users N_u

Average throughput is depicted in Fig. 2. A general observation is that both A4 and A5 exhibit a relatively constant behavior, regardless of the number of users belonging to the multicast group. It is also clear that A5 performs worse than A2, A3 and A4. However, the maximum difference between A2/A3 and A5 is about 5%, and as N_u increases the performance of all three algorithms converges. On the other hand, A4 shows an almost constant performance, which is always better than all other algorithms. In fact, as N_u increases, the achieved throughput decreases at a lower rate than A2 and A3.

Combining the performance in these two metrics, it is evident that A4 is the best choice when N_u is very high (above 200). However, this case is rather rare in practice. In more realistic situations, which metric is of major importance has to be decided. If resources should be used as efficiently as possible, A4 should be used, since it uses bandwidth in the most efficient way. On the other hand, if delay is critical for the service provided, A5 is the appropriate solution.

With respect to complexity, A4 is simpler than A5, since it does not require any additional feedback information from the users, apart from a simple NACK. In contrast, A5 requires from every user to add some information in the NACK, which makes it more complicated.

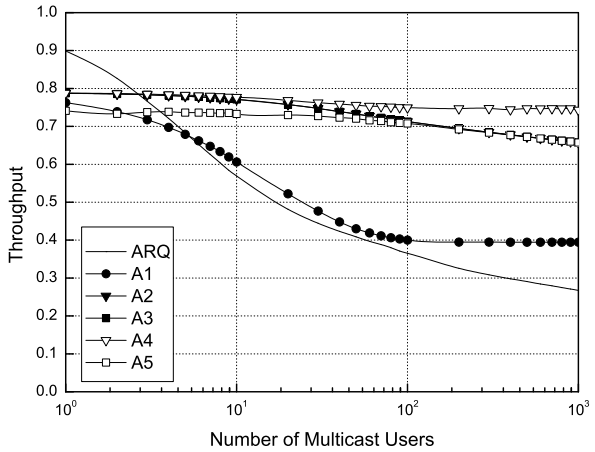

Fig. 2: Mean throughput as a function of the number of multicast users N_u

VI. Conclusions

In this letter, we proposed two adaptive HARQ algorithms for efficient and reliable multicast communications in UMTS networks. Our proposals present interesting results outperforming all the other algorithms proposed until today. The main idea behind the proposed algorithms is the introduction of a kind of memory into the system, which enables the RLC layer to make an estimation of the required redundant PDUs and thus to dynamically adjust the FEC mechanism in order to decrease the required retransmissions. In the future, we intend to examine the performance of our techniques using more detailed channel models. We also intend to make theoretical analysis of both delay and throughput performance.

References

[1] *Multimedia Broadcast/Multicast Service (MBMS): Architecture and functional description*, 3GPP Std. TS 23.246 v6.8.0, 2005.

[2] J. Nonnenmacher, E. Biersack, and D. Towsley, "Parity-Based Loss Recovery for Reliable Multicast Transmission," *IEEE/ACM Trans. Networking*, vol. 6, no. 4, pp. 349–361, Aug. 1998.

[3] H. Djandji, "An Efficient Hybrid ARQ Protocol for Point-to-Multipoint Communication and its Throughput Performance," *IEEE Trans. Veh. Technol.*, vol. 48, no. 5, pp. 1688–1698, Sept. 1999.

[4] C. Huitema, "The Case of Packet Level FEC," in *Proc. of IFIP 5th International Workshop on Protocols for High Speed Networks (PsHSN 1996)*, INRIA, Sophia-Antipolis, France, Chapman & Hall, Oct. 1996, pp. 109–120.

[5] C. Q. Yang, E. Hossain, and V. K. Bhargav, "On Adaptive Hybrid Error Control in Wireless Networks Using Reed-Solomon Codes," *IEEE Trans. Wireless Commun.*, vol. 4, no. 3, pp. 835–840, May 2005.

[6] M. Rossi, F. Fitzek, and M. Zorzi, "Error Control Techniques for Efficient Multicast Streaming in UMTS Networks: Proposals and Performance Evaluation," *Journal on Systemics, Cybernetics and Informatics*, vol. 2, no. 3, Nov. 2004.

[7] M. Rossi, M. Zorzi, and F. Fitzek, "Link Layer Algorithms for Efficient Multicast Service Provisioning in 3G Cellular Systems," in *Proc. of IEEE Global Telecommunications Conference 2004 (GLOBECOM'04)*, Dallas, Texas, USA, Nov. 2004, pp. 3855–3860.