RELIABLE MULTICAST FOR FLAT HIERARCHY NETWORKS BASED ON ADAPTIVE AIR CACHING

K. Manousakis, J. S. Baras

Center for Satellite and Hybrid Communication Networks
University of Maryland, College Park
College Park, Maryland

ABSTRACT

The evolution of satellite networks in the commercial and military world has pushed the research community towards the solution of important problems related to this kind of networks. One of those important problems is how to design an efficient reliable multicast protocol in a network where there is no hierarchy involved, and the link presents characteristics like high propagation delay and high BER. The existing reliable multicast protocols cannot be applied in the case of flat hierarchy networks, since those are based on intermediate receivers and local recovery techniques. So, we introduce the air caching technique, which serves as a fast access memory that is realized on the air and contains packets for the recovery of corrupted or erroneous data packets at the receivers. In this paper we present some of the protocols that we designed and are based on air caching combined with FEC and ARQ. In the past[7] we have presented protocols that are based on those techniques but the characteristics of the Air Cache were constant. In this paper we propose RM protocols, which are based on adaptive air caching, where the size and/or content of the Air Cache change dynamically based on feedback information. Our goal is to improve the delay and/or bandwidth usage characteristics of the non-adaptive Air Cache RM protocols.

INTRODUCTION

A lot of work has been done in the area of RM protocols in the recent years [1][2][3][4][6]. The majority of those protocols focus on networks with inherent topological or logical hierarchical architecture. The researchers have taken advantage of this hierarchy by developing Distributed Error Correction (DER) or Centralized Error Correction (CER) schemes that utilize techniques, which are based on local recovery and intermediate receivers (e.g., cache part of the transmitted packets between the source and the receivers, for fast recovery and filter requests for retransmission for ACK suppression). Those protocols have been proven successful for hierarchical terrestrial networks. Even though there is some work that partially involves hybrid networks (e.g., satellite and terrestrial), the researchers focus on the hierarchical terrestrial network and they do not take into consideration the characteristics of the satellite links (high BER, high propagation delay)[6]. In this work we make an attempt to design protocols that are customized for flat hierarchy networks and more specifically for satellite networks of one hop, where we need to invent techniques that compensate for the lack of hierarchy and present very promising performance characteristics. A technique that is of great importance and is used heavily in the protocols that we propose is called air caching. We dedicate part of the available bandwidth for continuously pushing critical packets for the fast recovery of the corrupted or erroneous data packets at the multicast receivers. Because of the continuous push of the Air Cache packets, these packets can be accessed faster for error recovery from the receivers, without issuing retransmission requests, which have to traverse the high propagation delay and high BER satellite link. Air caching is used to proactively correct errors at the receivers and improve the end-to-end delay. We have already presented the non-adaptive class [7] of reliable multicasting protocols that utilize air caching. In those protocols (UDPAC, RDPAC and PPAC) the Air Cache (e.g., size and content) is not adapted based on the progress of the reliable transmission of data. This work aims on the improvement of the performance (e.g., end-to-end delay and bandwidth usage) of the non-adaptive protocols by adapting the Air Cache.

In the next section we will present the flat network architecture. In the third section we will give an overview of the FEC and air caching methods along with a brief description of the non-adaptive class of protocols. In the fourth section we will present the adaptive class of protocols that we propose. In the fifth section we will give some performance evaluation results, related to the end-to-end latency of those protocols, as we collected them from the corresponding simulations. Finally, in section six we will conclude this paper.

NETWORK ARCHITECTURE

As we have already mentioned, the objective of this work is the design of reliable multicasting protocols for flat

0-7803-7625-0/02/$17.00

hierarchy networks, as the satellite networks are. The network architecture on which we based our study is described as a flat hierarchy or one-hop hierarchy. An example is given in the following figure.

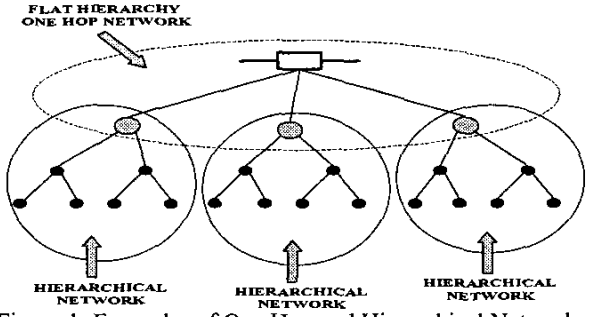

Figure 1: Examples of One Hop and Hierarchical Networks

The lack of hierarchy increases the difficulty of designing an efficient reliable multicasting protocol by using techniques like ARQ, and FEC combined with local recovery and caching some of the transmitted packets at intermediate points. For the latter reason we introduced the concept of air caching, since its operation fits the framework of the broadcast one-hop link. Details about air caching and how is being used for the design of RM protocols follow.

FEC, AIR_CACHING AND NON-ADAPTIVE AIR CACHE PROTOCOLS

The protocols that we propose are based on air caching combined with techniques that have been successfully applied in the area of reliable multicasting such as Forward Error Correction (FEC) and Automatic Repeat Request (ARQ). The latter two techniques are well known, as opposed to air caching, which we will introduce in a while. Also, we will briefly describe FEC and the non-adaptive Air Cache RM protocols that we had proposed [7].

A. Forward Error Correction (FEC)

FEC is a proactive mechanism, as opposed to the reactive nature of ARQ, where the retransmissions are taking place only when the receivers are requesting them. FEC is based on the transmission of parity packets along with the data packets. The generation of the parity packets is based on the data packets to be transmitted (TG). One of the most important and attractive characteristics of FEC is that each parity packet can correct any data packet at the receiving end. Assume that the TG has size k and h parity packets have been generated, then the TG is considered to be delivered reliably at the receiving end when k out of the $k+h$ packets have been received correctly, and that is because each parity packet can compensate for any erroneous or corrupted data packet [6].

B. Air Caching

We can characterize Air Cache [5] as a continuous broadcast or continuous push of data. The immediate result of broadcasting the data continuously is that these data can be accessed from the end hosts more frequently and with less average end-to-end latency compared to non-continuously pushed data. We will give a general description of air caching operation, by looking on the following figure:

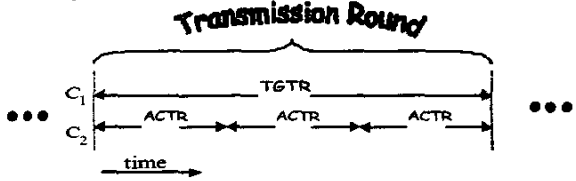

Figure 2: Example of the Air Cache operation

There are two dedicated channels for transmission, the C_1 and C_2. The C_1 is used for the transmission of the data packets to be delivered reliably at the multicast group. The C_2 is the so-called Air Cache, which contains two different types of packets depending on the error recovery technique that the corresponding protocol is relying on. If the protocol uses FEC then the Air Cache contains parity packets, otherwise if the protocol is based on ARQ then the Air Cache contains copies of a subgroup of the data packets that are being transmitted concurrently in C_1. The time interval required for the transmission of the data packets in C_1 is called Transmission Group Transmission Round (TGTR), and the time interval required for the transmission of the packets in the Air Cache is called Air Cache Transmission Round (ACTR). The relation of the latter two parameters is:

$$ACTR \leq TGTR$$

The technique of air caching has never been used in designing RM protocols and as we are going to prove there is a significant boost in the performance of those protocols.

C. Non-adaptive Air Cache RM Protocols

The class of non-adaptive Air Cache protocols, which we had presented [7], includes the following protocols:

- UDPAC (Unchanged Data Packets in the Air Cache)
- RDPAC (Replace Data Packets in the Air Cache)
- PPAC (Parity Packets in the Air Cache)

The names of those protocols reveal their main characteristics. The former two (e.g. UDPAC, RDPAC) use data packets in the Air Cache, so they use a combination of ARQ and air caching for the recovery of erroneous or corrupted data packets at the receivers. The difference between UDPAC and RDPAC is the refreshing rate of the Air Cache, but in both protocols the data

packets to be included in the Air Cache per ACTR are chosen randomly from the data packets that are transmitted in C_1. The combination of ARQ with air caching presents better performance characteristics than relying solely on ARQ. The PPAC protocol uses FEC combined with air caching for providing reliable multicast transmission. The Air Cache contains parity packets that are continuously transmitted per ACTR. Each parity packet can recover any data packet transmitted in C_1. In terms of required transmission rounds, PPAC presents very promising performance characteristics, much better than the corresponding performance of UDPAC and RDPAC. The advantage of UDPAC and RDPAC is that are lightweight protocols in terms of processing and memory requirements, in contrast to PPAC, where encoding and decoding is required because of the utilization of parity packets.

ADAPTIVE AIR CACHE PROTOCOLS

This work aims on the improvement of the non-adaptive class of RM protocols, in terms of bandwidth usage and/or required transmission rounds. There are three classes of adaptive Air Cache RM protocols. The classification is based on the fact that we can adapt the size of the Air Cache (i.e., for minimizing the bandwidth dedicated to the Air Cache) or the content of the Air Cache (i.e., for improving the delay performance) or both (e.g., size and content, in order to get the best performance with the *minimal* usage of network resources). The three classes of adaptive Air Cache RM protocols are:

- Content Adaptation of Air Cache RM Protocols
- Size Adaptation of Air Cache RM Protocols
- Hybrid Adaptation of Air Cache RM Protocols

For each one of the above classes we propose a reliable multicasting protocol. Namely, those protocols are:

- Adaptive Content Data Air Cache (ACDAC)

ACDAC's corresponding static Air Cache protocols are the RDPAC and UDPAC. In ACDAC, using the feedback in each transmission round we adapt the contents of the Air Cache, as opposed to UDPAC and RDPAC, in order to achieve better performance than the latter protocols.

- Adaptive Size Parity Air Cache (ASPAC)

ASPAC's corresponding static Air Cache protocol is the PPAC protocol (e.g., both use parity packets in the Air Cache). The adaptation is in terms of the size of Air Cache. The goal here is to design the adaptive version of PPAC with the same levels in performance but with the minimal Air Cache bandwidth usage.

- Hybrid Adaptive Data Air Cache (HADAC)

HADAC's corresponding static Air Cache protocols are the UDPAC and RDPAC protocols. The adaptation takes place both on content and size. Actually, HADAC is an extended version of ACDAC. Our goal is to improve the delay performance of the corresponding static Air Cache protocols, like in ACDAC, but with the minimal bandwidth usage.

A. Adaptive Content Data Air Cache (ACDAC)

Basic Algorithm

Round 1: Transmit concurrently the data packets in C_1 and the randomly chosen Air Cache contents in C_2.

Round K (K>1): If there are no requests for retransmission then the reliable transmission of the TG data packets has been completed in *K-1* rounds and the algorithm stops. If there are requests for retransmission then there are new transmissions on C_1 (e.g. TG data packets) and C_2 (e.g., adapted Air Cache contents – the adaptation algorithm follows). We increase K ($K=K+1$) and we repeat *Round K*.

Adaptation Algorithm of the Air Cache

Round 1: The Air Cache is filled with randomly chosen data packets from the TG.

Round K (K>1): Taking into consideration the feedback, the Air Cache is not anymore filled with randomly chosen data packets. Assume that the distinct requested data packets are *DRQpacks* and the Air Cache size per round is *ACRsize*. There are two different cases that result in updating the Air Cache in two different ways.

Case 1: if ($DRQpacks \geq ACRsize$) then we choose the *ACRsize* packets that have been requested more for retransmission from the participants of the multicast group in each transmission round.

Case 2: if ($DRQpacks < ACRsize$) then we fill the Air Cache with every requested packet (e.g., *DRQpacks*) for retransmission. The remaining ($ACRsize - DRQpacks$) is filled with the $ACRsize - DRQpacks$ most requested data packets of the previous transmission round. In this case, we could have resized the Air Cache but we assume that the size of Air Cache remains constant in each ACTR.

B. Adaptive Size Parity Air Cache (ASPAC)

Basic Algorithm

Round 1: We have two concurrent transmissions going on. In C_1 we have the transmission of the TG data packets and in C_2 we have the transmission of the *ACRsize* parity packets. For this initial round we fill the Air Cache with the maximum number of parity packets (e.g. max(*ACRsize*) – max(Air Cache Round size)).

Round K (K>1): If there are no requests for retransmission then all the members of the multicasting group have received the TG data packets correctly, so the reliable transmission is completed in *K-1* transmission rounds and

the algorithm stops. If there are requests for retransmission then there are new transmissions on C_1 (e.g. TG data packets) and C_2 (e.g., parity Air Cache whose size has being adapted – the adaptation algorithm follows). We update K ($K=K+1$) and we repeat *Round K*.

Adaptation Algorithm of the Air Cache

Round 1: Air Cache is filled with the max number of parity packets that can be transmitted in each Air Cache Transmission Round. We assume that this number is max(*ACRsize*).

Round K (K>1): if we assume that the distinct packets of the TG that have been requested for retransmission in round *K-1* is $DRQpacks_{K-1}$ and the maximum number of packets that the Air Cache can contain is max(*ACRsize*), then the size of the Air Cache in round K is $ACRsize_K$ and is specified from the following formula:

$$ACRsize_k = \min(DRQpacks_{K-1}, \max(ACRsize))$$

C. Hybrid Adaptive Data Air Cache (HADAC)

Basic Algorithm

Round 1: We have two concurrent transmissions going on channels C_1 and C_2. In C_1 there is the transmission of the TG data packets and in C_2 the transmission of the Air Cache, which contains max(*ACRsize*), randomly chosen, data packets from the transmission group (TG).

Round K (K>1): If there are not requests for retransmission then the transmission of the TG data packets is completed, because all the members of the multicast group have received the data correctly. In this case the number of the required transmission rounds for the reliable delivery of TG data packets is *K-1* and the algorithm stops. If there are requests for retransmission then there are new transmissions on C_1 (e.g. TG data packets) and C_2 (e.g., data Air Cache, whose contents and size have being adapted – the adaptation algorithm follows). We increase K ($K=K+1$) and we repeat *Round K*.

Adaptation Algorithm of the Air Cache

Round 1: The Air Cache is filled with the max number of packets (e.g., max(*ACRsize*)) that can be transmitted per ACTR. The contents of the Air Cache are randomly chosen data packets from the TG (e.g., like in ACDAC).

Round K (K>1): Based on the feedback from the multicast receivers, the content and size of the Air Cache are adapted properly. The size is adapted based on the same idea, which is used in ASPAC. If we assume that the distinct packets, which have been requested for retransmission in *Round K-1* are $DRQpacks_{K-1}$ and the maximum allowable number of packets in the Air Cache is max(*ACRsize*), then the size of the Air Cache in *Round K* (e.g. $ACRsize_K$) will be:

$$ACRsize_K = \min(DRQpacks_{K-1}, \max(ACRsize))$$

The contents of the Air Cache are adapted as follows:

In *Round K* the Air Cache will contain:

Case 1

(if $ACRsize_K = \min(DRQpacks_{K-1}, \max(ACRsize)) = \max(ACRsize)$)

The $\max(ACRsize)$ most requested data packets.

Case 2

(if $ACRsize_K = \min(DRQpacks_{K-1}, \max(ACRsize)) = DRQpacks_{K-1}$)

The Air Cache will have size $DRQpacks_{K-1}$ and will contain every requested ($DRQpacks_{K-1}$) data packet from *Round K-1*.

PERFORMANCE EVALUATION

In this section we will present some of the important results that we collected by simulating the adaptive Air Cache RM protocols. We will focus mostly on the performance of the protocols in terms of the required transmission rounds for the reliable transmission of a group of data packets (TG). Also, we will exploit the bandwidth savings in ASPAC, because of the size adaptation of the Air Cache.

In figure 3 we compare the performance of the adaptive and non-adaptive protocols. The performance comparison is done in terms of the required transmission rounds for the reliable transmission of 20 data packets (TG=20) for various Air Cache sizes, varying from 0 packets (e.g., RM protocols without Air Caching) to 10 packets.

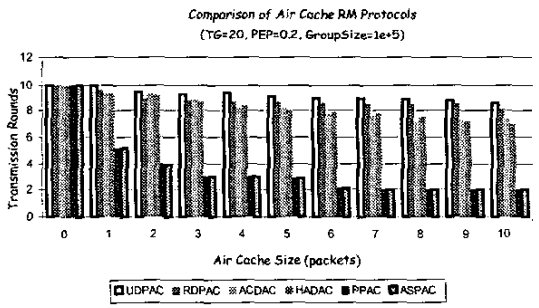

Figure 3. Adaptive vs. non-adaptive Air Cache RM protocols (TG=20 data packets, PEP=0.2, Group Size =10^5 hosts)

A first observation is that the protocols that are based on FEC and air caching perform much better than the protocols that combine ARQ and air caching, a result that was expected. Also, the performance of the protocols, which utilize air caching, is better compared to the case where there is no utilization of this technique (*ACRsize=0*). On the other hand, the performance improvement is saturated as we increase the Air Cache size after a threshold point. By observing and analyzing

the results from the comparison of adaptive and non-adaptive protocols we reached some very important conclusions:

- Even though the size of Air Cache is adapted in ASPAC, which results in using overall less bandwidth per transmission round (see Figure 4), there is no degradation in performance compared to PPAC, which uses constant Air Cache size.
- ACDAC presents better performance characteristics compared to its non-adaptive counterparts (e.g., UDPAC, RDPAC), because of the content adaptation of the Air Cache.
- HADAC compared to its non-adaptive counterparts (UDPAC, RDPAC) presents better performance characteristics because of the content adaptation of the Air Cache.
- HADAC and ACDAC have similar performance characteristics, because both of them adapt the content of the Air Cache. On the other hand, HADAC uses lesser bandwidth than ACDAC because of the combined content and size adaptation.

In the following graph we present results that have to do mainly with ASPAC protocol and the size adaptation of the Air Cache. As we have already mentioned PPAC (the non-adaptive counterpart of ASPAC) uses constant Air Cache size throughout the reliable transmission. The question that arises is if the adaptation of the Air Cache size results in bandwidth savings without degradation in performance compared to PPAC. The graphs in figures 3 and 4 are being used for drawing an answer to this interesting question.

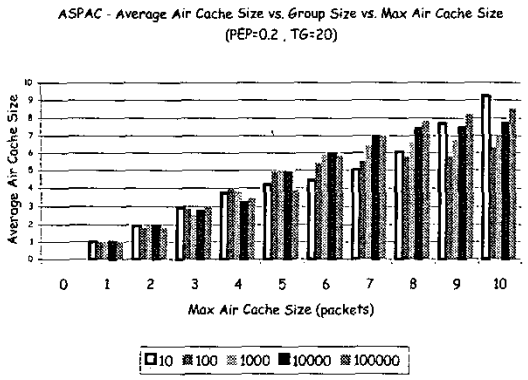

Figure 4. Avg. size vs. the max size dedicated to Air Cache. (TG=20, PEP=0.2)

Observing carefully the above graph we can reach to some very important conclusions since the average savings on Air Cache bandwidth do not vary smoothly for various group sizes and various maximum Air Cache sizes. Instead, as we vary the maximum Air Cache size and the multicast group size, the results do not follow a standard pattern but can be drawn in accordance to the varying parameters of the protocol (i.e., group size, max Air Cache size). The important point, which is drawn from the previous two graphs, is that the behavior of the ASPAC protocol in terms of the required transmission rounds remains almost unchanged compared to PPAC performance, despite the fact that the average Air Cache size used is always less or equal to the constant Air Cache size used from the PPAC protocol per ACTR. Based on the latter observations, it is obvious that ASPAC retains the promising performance characteristics of the PPAC protocol using less extra bandwidth.

CONCLUSIONS

The goal of this work was to improve the non-adaptive Air Cache protocols that we presented in [7]. The improvement process focuses on the performance characteristics and the average bandwidth usage of the protocols. Based on the results that we presented above, the protocols that we propose are an improved version of their non-adaptive counterparts. The adaptation in size improves the average bandwidth usage of the protocol without degradation in performance, compared to the protocols that do not adapt the Air Cache size. The adaptation in content improves the performance of the protocols in terms of the required transmission rounds for the reliable transmission of a group of data packets. The latter facts prove the improvement of the adaptive protocols compared to the non-adaptive ones, in terms of the delay performance and the usage of network resources.

REFERENCES

[1] S. Floyd, V. Jacobson, C-G. Liu, S. McCanne, L. Zhang, "A Reliable Multicast Framework for Light-weight Sessions and Application Layer Framing", IEEE Transactions on Networking, November 1996

[2] C.K. Miller, "Multicast Networking and Applications", Addison Wesley Longman, Inc., 1999

[3] D. Rubenstein, S. Kasera, D. Towsley, J. Kurose, "Improving Reliable Multicast Using Active Parity Encoding Services", Proc. of INFOCOM '99

[4] J. Nonnenmacher, E.W. Biersack, D. Towsley, "Parity-Based Loss Recovery for Reliable Multicast Transmission", IEEE/ACM Transactions on Networking, Vol. 6, No. 4, August 1998

[5] K. Stathatos, N. Roussopoulos, J.S. Baras, "Adaptive Data Broadcasting Using Air Cache", WOSBIS, 1996.

[6] S. M. Payne, J. S. Baras, "Reliable Multicasting via Satellite: Delay Considerations", Proc. of ATIRP 2000

[7] K. Manousakis, J. Baras "Reliable Multicast over Satellite",Proc. of ATIRP 2001