Scalable Architecture for Multimedia Multicast Internet Applications

Tatiana Polishchuk
Helsinki Institute for Information Technology HIIT
Aalto University, Finland
Email: tatiana.polishchuk@hiit.fi

Michael Karl
Intel Visual Computing Institute
Saarland University, Germany
Email: karl@intel-vci.uni-saarland.de

Thorsten Herfet
Telecommunications Chair
Saarland University, Germany
Email: herfet@cs.uni-saarland.de

Andrei Gurtov
Helsinki Institute for Information Technology HIIT
Aalto University, Finland
Centre for Wireless Communications, University of Oulu, Finland
Email: gurtov@hiit.fi

***Abstract*—We propose a scalable multicast architecture for potentially large overlay networks. Our techniques address suboptimality of the adaptive hybrid error correction (AHEC) scheme in the multicast scenarios. A hierarchical multi-stage multicast tree topology is constructed in order to improve performance of AHEC and guarantee QoS for the multicast clients. The multicast tree is divided into subtrees, called *regions*. Every region is assigned a *control node*, which serves the individual redundancy and retransmission requirements of the receivers within the region. Region sizes are bounded by the *maximum cost per region*, which defines the ability of the control nodes to serve the receivers of the assigned regions. We show that the multi-stage multicast architecture significantly reduces the amount of redundancy information introduced into the network and brings it closer to the Shannon bound.**

Keywords: Internet, multimedia, multicast, HEC, QoS

I. Introduction

Efficient and reliable transport of multimedia content is a critical issue of today's IP network design. The amount of this traffic type highly increased in the last years and it continues to grow [1]. But not only typical multimedia traffic originated by web platforms like Youtube is dominating the IP based world. A quickly growing interest to the real-time streaming and interactive applications (e.g. IPTV, online gaming, remote surgery) leads to the higher stressed networks. The major differences between real-time and non real-time (*elastic*) traffic are the intrinsic speed behavior and strict error-rate limit of the real-time traffic. Elastic traffic can tolerate fluctuations during transmission up to a certain degree, whereas real-time flow traffic cannot. New transmission approaches have to be investigated to prevent the *Future Media Internet* from being impaired by non-essential traffic. Owing to error-prone network segments the use of new error-correction schemes are also required, e.g. *hybrid error correction* frameworks.

The primary focus of this work is design and evaluation of a scalable multicast overlay architecture for a wide range of multimedia Internet applications. We propose an effective treelike hierarchical structure, which is adapted to the underlying network topology and provides a highly efficient multimedia transmission when used by the error-correction scheme, called AHEC (Adaptive Hybrid Error Correction). AHEC [5] operates according to the Predictable Reliability under Predictable Delay (PRPD) paradigm: based on a statistical channel model it adds the optimal amount of redundant information to meet the desired residual packet loss rate (PLR) within limited time constraint. The flexible combination of limited packet retransmission issued by negative feedback and adaptive, packet-oriented Forward Error Correction (FEC) spans a large parameter space with few feasible configurations: the number of retransmission rounds and the FEC block length share the overall time budget.

Scalability of the multicast dissemination trees controlled by the hybrid error correction has never been addressed before, to the best of our knowledge. We developed an algorithm for optimal placement of the control nodes within the multicast tree, which serve individual redundancy and retransmission requirements of the receivers within the subtrees called regions. Such an architecture enables a hierarchical error correction that efficiently isolates error-prone regions within the network and thus protects nearly error-free regions from being flooded by redundancy introduced by high error-rate sections. This approach naturally reduces the network load in real-time multicast scenarios with heterogeneous network structures. We discovered that the needed amount of redundancy information for traffic protection introduced by the AHEC scheme was significantly reduced. To evaluate the performance of the proposed architecture we define *proximity* metric and show that the amount of required redundancy information is now closer to the theoretical Shannon limit.

The reminder of this paper is structured as follows. Section II reviews the related work. Section III introduces the general multi-stage multicast architecture. Section IV reviews notations and formulas for redundancy information and provides an efficiency metric to benchmark the proposed architecture. Section V presents the experimental results. Section VI outlines the future work and concludes the paper.

II. Related Work

The idea of optimizing performance of an error correction scheme by using a hierarchical tree structure was first introduced by Radha and Wu in [10] and [15]. They developed a recursively optimal scheme for the placement of a given number of network-embedded FEC codecs within a randomly generated multicast network with known link loss rates.

Shan et al. proposed in [11] an overlay multi-hop FEC (OM-FEC) scheme that provides FEC encoding and decoding capabilities at the intermediate nodes in the overlay path. Based on the network conditions, the end-to-end overlay path is dynamically partitioned into segments and appropriate FEC codes are applied over those segments.

Kopparty et al. [7] and Paul et al. [9] introduced the idea of optimizing the lengths of retransmission rounds in the design of transport multicast protocols (SplitTCP and RMTP), by allowing buffering at some intermediate nodes.

Karl et al. [4] state that there exists a saturation point for the number of intelligent intermediate nodes on a network path and propose a distributed solution approach based on the Multiple-Choice Knapsack Problem (MCKP) in [5]. A decentralized mechanism to effectively split a network path according to its physical properties was described in [6]. Gorius et al. [3] developed a domain separation approach for hybrid error correction schemes applied to the real-time multimedia streams.

Tan et al. [12] optimized the performance of AHEC scheme for DVB services over wireless home networks. The authors analysed the needed redundancy information for the HEC-PR and HEC-RS cases of the general AHEC frameworks and showed how to minimize RI value in multicasting scenarios with small group sizes (about 7 receivers). Later in [13] they noticed that needed RI of the general AHEC architecture grows quickly in the multicast scenarios with the increase of the group size if the size of the group is small (less than 20 receivers), but if the size of the group is large enough, the total needed redundancy will increase very slightly with the increase of the group size. They concluded that AHEC scheme can be suitable for the multicast scenarios with large groups. Previous work considered only limited multicast topologies with small groups of multicast receivers connected directly to the source. We extend the multicast scenario to the more realistic Internet tree structure and study how to optimize AHEC for a wider range of multimedia multicast applications and bigger groups.

III. The Scalable Multicast Multistage Architecture

In this paper we design a *scalable multicast multistage architecture* targeting to optimize performance of the adaptive hybrid error correction (AHEC) scheme in order to serve predictable reliability for the needs of multimedia Internet applications. The architecture has to adhere to the two strict constraints defined by the application: maximum allowed retransmission delay and target error rate at the receivers.

Consider a general network $\mathcal{G}$. One node S of the network is the *sender*. A set R of nodes are the *receiver* nodes. Each link in $\mathcal{G}$ is characterized by its round trip time (RTT) and packet loss rate (PLR). We integrate the parameters of the link into a single number that we call the link's *cost*, which roughly defines the redundancy level required to serve the link. (We will explain how the cost is calculated in further sections.) Based on the costs, we construct the tree $\mathcal{T} = (V, E)$ of shortest paths from the main sender S to all receivers in R.

A. Control Nodes and Regions

We divide the multicast tree $\mathcal{T}$ into subtrees, called *regions*. The root of the region is called the *control node* of the region. It serves the individual redundancy and retransmission requirements of the receivers within the region.

Region *size* s_j is the longest distance in terms of costs from the control node to any receiver within this region. It should not exceed the *maximum cost per region*: $s_j < c_{max}$, which defines the ability of control nodes to serve the receivers of the assigned regions. The distance from the source S to the control node C_j, is obviously always shorter than the distance from C_j to any receiver R_i which belongs to the region, controlled by C_j: $|SR_i| > |C_jR_i|$.

B. Control Node Functionality

Control node receives data from the source, stores the current in-flight data in a buffer, decodes the content and forwards data directly to the end receivers. Control node also retransmits missing data segments upon request of individual receivers within its region. Furthermore, it is possible to isolate *bad* receivers and limit their influence on the nodes outside of their regions. One control node can control either a large number of receivers connected to the node through good quality (i.e. low cost) links, or fewer receivers connected through links with worse quality (i.e. higher cost), or a combination of different types of receivers.

The overhead of the insertion of control nodes is in fact negligible. The router assigned to be a control node stores only a small amount of data packets in flight to be able to serve retransmission requests from the receivers. Obviously modern routers possess sufficient amount of memory to store a several seconds the data from the multicast data stream. The decoding times are also small in comparison to the end-to-end paths RTT values.

C. Control Nodes Assignment Algorithm

The algorithm finds optimal placements for control nodes needed to serve all the receivers minimizing the total number of control nodes.

Let's define a receiver R_i to be *served* if it has a control node C_j within the distance c_{max} from it. The control node is called *critical* if it lies at the maximum distance from R_i.

The algorithm is scanning through the tree from the leaves up to the root starting from the most remote receiver. It is consequently checking whether the condition $|Rn_i| \leq c_{max}$ is satisfied for the current node n_i. If for the following node $|Rn_{i+i}| > c_{max}$ (or if we reached the tree root), then n_i is the critical node for the receiver R and n_i is assigned to be a control node C_j. The whole subtree with the root in C_j is assigned to be the $j - th$ region and is removed from further consideration. The most remote receiver is to be found in the rest of the tree and the scanning repeats. The procedure stops when each receiver of the original multicast tree is served. A pseudo code of this algorithm is provided in Algorithm 1.

Algorithm 1 Control nodes assignment algorithm

```
1:  for all receivers in the tree do
2:      find distances from the root to all the receivers
3:      find $R_j$ - the most remote receiver
4:      $n_i$ - current node
5:      while $n_i \neq root$ do
6:          go up the tree
7:          find the critical node for $R_j$ and assign it to be a
            control node $C_j$.
8:          remove subtree with the root in $C_j$ from the tree
9:      end while
10:     $j = j + 1$
11: end for
```

Next we prove the optimality of the proposed algorithm. Let l be the most remote receiver. Let c be the first control node found by the algorithm. In any feasible solution, the subtree with the root in c must contain a control node c'. Otherwise l is not served. We claim that without loss of generality, c' can be shifted to c. Indeed, suppose that shifting c' to c makes some leaf l' unserved. This means that $|l'c| > c_{max}$. But we know that $|lc| \leq c_{max}$, and for the main source s we have $|l's| = |l'c| + |cs| > |lc| + |cs| = |ls|$, which contradicts to the assumption that l is the furthest receiver. We conclude that since on each iteration the algorithm starts from the most remote receiver l, the distance to any other node in the subtree with root in c will be less than $|cl|$.

The algorithm is fast with the running time $O(n \log m)$, where $m \log m$ is required to sort m receivers. Note that in the Internet-like topologies for which the algorithm is designed, the number of receivers is regularly less than half of the total number or tree nodes.

IV. Redundancy optimization

Inspired by the previous work [12] we continue working on optimizing the amount of redundancy information required by AHEC to better serve the needs of particular multimedia applications. The total needed redundancy for each receiver depends on the number of retransmission rounds, which in turn has a direct connection to the end-to-end RTT and PLR values. In the multicast scenario the amount of redundancy is dictated by the worst receiver in the group. By introducing control nodes between the source and end receivers we shorten the RTT needed for retransmissions and isolate the links with high loss probability so that they only influence the performance of the corresponding region, but not the whole multicast tree.

A. Redundancy Information

Further we review the definition and formulas we use to calculate RI_{HEC} according to the framework provided in [12].

Definition 1. *Redundancy information (RI) is the controlled redundancy added by the channel encoder and required to protect the receiver from any errors during data transmission.*

In the general AHEC framework [14] the redundancy information consists of two parts. One part, denoted by r, is delivered with the main data block during the first transmission and it is produced by the FEC component of AHEC. The overall transmitted block length is calculated as $n = k + r$ for the data size of k and the redundancy amount of r. The second part of the total redundancy is the data, which restores lost packets after retransmissions. The number of retransmissions available is limited by the delay budget, which is left after the first transmission.

In this current work we focus on optimizing the second part of redundancy information produced by retransmissions, since optimization of FEC with hierarchical tree structures was already addressed in the related work ([10], [15], [11]).

In what follows we calculate RI_{HEC}, the redundancy information amount required by the HEC-PR scheme, which is a degenerate case of the general AHEC architecture [14]. In HEC-PR the number of source data packets in one encoding block $k = 1$, and the architecture acts as a pure ARQ-based scheme. The redundant packets during all retransmissions are always the copies of the source data packets.

We use the following notation:
P_{target} - target packet loss rate requirement;
D_{target} - target delay requirement;
RTT_i - round trip time of the link;
P_i - packet loss rate of the link;
RTT_{e2e} - round trip time of the end to end path between the source and receiver;
P_{e2e} - original end-to-end PLR;
T_s - average interval between two continuous data packets;
N_T - the maximum possible number of transmissions for each data packet;

N_{rr} - the maximum possible number of retransmission rounds for each data packet;
$\tilde{N_{rr}} = \min(N_{rr}, N_T)$ - maximum allowable number of retransmissions (practical);
N^q - the number of transmissions of each missing data packet during q-th retransmission round;
RI_{HEC} - the total redundancy information.

Formulas for the RI_{HEC} calculation:
$RTT_{e2e} = \sum_{i=1}^{N} RTT_i$; $P_{e2e} = 1 - \prod_{i=1}^{N}(1 - P_i)$
$N_T = \lceil \frac{\log(P_{target})}{\log(P_{e2e})} \rceil$; $N_{rr} = \lfloor \frac{D_{target} - \frac{RTT_{e2e}}{2} - (N_T - 1) * T_s}{RTT_{e2e} + T_s} \rfloor$

$$RI_{HEC} = \sum_{q=1}^{\tilde{N_{rr}}} N^q P_{HEC}(q-1), \text{ where } P_{HEC}^{q-1} = P_{e2e}^{\sum_{q=0}^{q-1} N^q}$$

We are minimizing the total needed redundancy information by optimizing the number of retransmission rounds needed to provide P_{target} without D_{target} violation:
$RI_{HEC}^{opt} = \arg\min RI_{HEC}$, s.t. $1 \leq N_{rr} \leq \tilde{N_{rr}}$

B. Shannon limit

Shannon's coding theorem [8] defines the theoretical maximum information transfer rate of the binary erasure channel with the error probability P to be equal to $1 - P$. Therefore, for reliable data transmission the amount of redundancy information should be at least $RI_{lim} = \frac{P}{1-P}$.

Obviously, in practice we need slightly more than this theoretical value. Our goal is to minimize the redundancy information amount used in the error correction scheme.

C. Proximity

Optimization brings redundancy information amount closer to the theoretical lower bound. In order to compare performance of the AHEC scheme for different network settings we introduce the following metric:

Definition 2. *Proximity ϵ is the difference between the amount of redundancy information introduced into the network by the AHEC scheme for the current system setup and the desired optimal value of the redundancy required for that system according to Shannon bound: $\epsilon = RI_{HEC} - RI_{lim}$.*

Proximity takes non-negative values only and shows how close AHEC scheme approximates the optimum.

D. Multi-hop redundancy

In a traditional end-to-end scenario the path redundancy is sent across several physical or virtual path segments. The average end-to-end path redundancy is $RI_{e2e}^{avg} = RI_{HEC}$. If we consider a multi-hop scenario, we have to calculate the redundancy $RI_{HEC,i}$ for each segment i along the path. Assuming a path with N segments, the average multi-hop path redundancy is $RI_{mh}^{avg} = \frac{\sum_{i=1}^{N} RI_{HEC,i}}{N}$. Note, that RI_{lim}^{avg} is calculated analogously to RI_{mh}^{avg}. In the following we use the average redundancy values for proximity calculation.

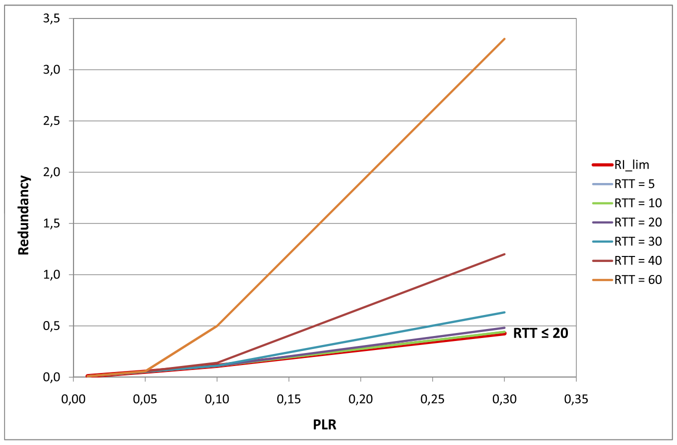

Fig. 1. RI sensitivity to the changes in RTT.

E. Cost Metric

Cost metric is a crucial factor for the optimization process since it directly relates the network characteristics to the mathematical calculations. We experimented with several cost metrics to test the effectiveness of our multi-stage multicast scheme: 1) costs equal to RTT values, 2) costs equal to PLR values or 3) costs equal to $RTT \cdot PLR$, and studied the sensitivity of the amount of redundancy to each of them. Each metric demonstrated the influence on the results of optimization procedure. And the most noticeable dependency was observed between the first metric (costs equal RTT values) and the amount of redundancy in the HEC-PR scheme.

Figure 1 illustrates how the amount of the HEC-PR redundancy reacts to the changes in RTT value for a fixed PLR range. The initial network parameters were set as described in [12]. If we take a point with the fixed $PLR = 0.1$ and the corresponding optimal redundancy $RI_{lim} = \frac{PLR}{1-PLR} = 0.1(1)$, then setting $RTT = 20ms$ brings RI_{HEC} quite close to the RI_{lim}, with the proximity $\epsilon \leq 0.0001$. Further reduction of RTT will not significantly improve the RI_{HEC} value, but increase the complexity of network management.

We discovered that for each bounded range of PLR in the network we can find an appropriate threshold RTT value, such that the corresponding HEC scheme results in the redundancy close to the optimal with a certain fixed proximity. This showed us the way how to choose the right c_{max} value corresponding to the desired proximity ϵ_{target}.

F. Problem Statement

The redundancy optimization problem can be described as follows.

Given a fixed network topology with known PLR and RTT values of all links and a given multicast scenario (ex. HDTV) with known data rate, target delay D_{target} and residual error rate P_{target}, find the minimum value of c_{max} in order to bring the redundancy value to within ϵ_{target} to the Shannon limit.

We are not aiming to reach the optimal Shannon bound for each case, but to find the appropriate limiting cost values, which guarantee the redundancy is close to the optimum with the given proximity ϵ_{target}.

Next we assign the control nodes according to the proposed c_{max}. The number of control nodes is a crucial factor. Each control node splits the transmission path into segments. Each segment must be assigned a smaller portion of the overall end-to-end D_{target} budget which obviously leads to higher redundancy on these segments. In [4] the authors showed that there is a saturation point for the number of segments up to which the overall redundancy performance increases. Beyond this saturation point the error correction performance is decreasing. Therefore, too many control nodes downgrade the redundancy optimization.

V. Empirical evaluation

In order to benchmark the efficiency of the proposed redundancy optimization scheme we conducted a series of simulations with different tree topologies and initial parameter sets. The redundancy optimization procedure goes in several iterations. Initially we specify the application-driven parameters, such as D_{target}, P_{target}, the desired proximity to Shannon bound ϵ_{target} and the ranges for RTT and PLR values of the tree links. An example of system parameters used in our simulations can be found in Table I. The chosen low values for P_{target} and D_{target} are required for such a demanding applications as DVB services or gaming.

Multicast trees were generated with random link parameters within the specified ranges, but the tree topologies were designed in the way that they adhere to the power-low distributions typical for the multicast trees extracted from the real Internet topologies as described in [2].

A. The Procedure

Step 1. For the given tree we identify the worst receiver with respect to the $cost = RTT$ metric, calculate the required amount of redundancy RI_{HEC} and the resulting proximity ϵ. If the proximity is greater than the desired ϵ_{target} we start optimizing the amount of redundancy by introducing the control nodes into the tree. For the fixed PLR_{e2e} of the worst receiver and variable RTT values we find the threshold, after which proximity starts to grow away from zero, which gives us the c_{max} value. Next we apply the control node assignment algorithm described in Section III-C with respect to the optimum c_{max} value. The resulting control nodes break the end-to-end paths to the receivers into several segments. We recalculate the multi-hop redundancy RI_{HEC} for the worst path as an average among the segments together with the corresponding proximity. If the new proximity ϵ' is closer to the desired ϵ_{target} than the initial one, we proceed to the next iteration. Otherwise the procedure stops, we conclude that for the given setup it is not possible to achieve the desired ϵ_{target} and report the best proximity we have achieved, the corresponding number of control nodes and recommend their optimal placement.

Step 2. For the second step of optimization procedure we reconstruct the tree in the following way: the control nodes of the regions obtained on the previous iteration are now replacing the whole region and the other nodes assigned to the region (if they are not control nodes for the other regions) are relaxed from further consideration. We also adjust the D_{target} value by subtracting the maximum cost among the regions. This operation reflects the fact that some part of D_{target} was consumed inside the lower regions, which in the worst case corresponds to the cost of the worst region. Next, we repeat all the steps as described for the *Step 1.*

After each iteration the tree recursively shrinks and converges to the root. The optimization procedure stops when either the desired proximity ϵ_{target} is achieved or if after a certain iteration the proximity is not improving, which means it is not possible to optimize the redundancy further.

B. Numerical Example

In the following numerical example with a 10-node tree we demonstrate how the proposed optimization works step by step. Initial system parameters are specified in Table I. Figure 2 illustrates the three iterations of the redundancy optimization procedure, after which the desired proximity ϵ_{target} was successfully reached. The results of the optimization procedure after each iteration are presented in Table II.

First we calculate the starting values for RI_{lim} and RI_{HEC} for the worst receiver in the tree, which is node 7 in this case. The corresponding proximity value is $\epsilon = 0.0339$ and is obviously greater than the desired $\epsilon_{target} = 10^{-5}$. Next the optimal c_{max} value is calculated for further redundancy optimization. At the first iteration the control node assignment algorithm identifies two control nodes: 1 and 5. The costs of the corresponding regions are calculated and the maximum of these values is subtracted from the total D_{target} for further optimization. Both control nodes lie on the end-to-end worst path (Figure 2 - a) and separate the path into three segments. The multi-hop redundancy values are calculated and, as can be seen in the first column of Table II, the resulting proximity at the end of the first iteration was reduced to $\epsilon' = 0.0056$. It is still greater than the target proximity $\epsilon_{target} = 10^{-5}$, and we proceed to the second iteration. The tree is reconstructed and the redundancy is further optimized in the same manner with the new D_{target}. After the third iteration the corresponding proximity finally reached the desired ϵ_{target}.

TABLE I
Simulation parameters

P_{target}	D_{target}	ϵ_{target}	RTT_{link}	PLR_{link}
10^{-6}	$200ms$	10^{-5}	$10 \dots 50$ ms	$10^{-3} \dots 10^{-2}$

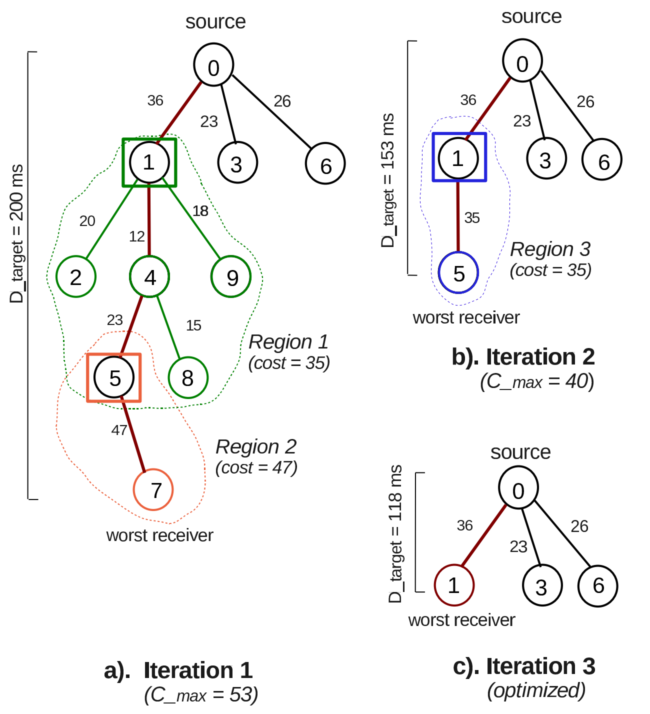

Fig. 2. Example of redundancy optimization procedure for a 10-node tree.

TABLE II
EXPERIMENTAL RESULTS

Parameter	Iteration 1	Iteration 2	Iteration 3
Worst receiver	7	5	1
RI_{lim}	0.0173	0.0108	0.0068
RI_{HEC}	0.0513	0.0323	0.0068
ϵ	0.0339	0.0214	0
D_{target}	200ms	153ms	118ms
c_{max}	53	40	-
Control nodes	1, 5	1	-
Max region cost	47	35	-
ϵ'	0.0056	0.0053	10^{-6}

C. Experiments

Further we conducted a series of experiments with different tree sizes. For each tree size the experiment was repeated one hundred times. We applied the redundancy optimization procedure and measured the resulting proximity for each case. Figure 3 illustrates average relative improvement of proximity for each of the tree sizes. As you can see in each case the proximity was significantly improved, which means that the total amount of required redundancy information was reduced, and in fact in some cases it achieved the theoretical Shannon bound. As we mentioned before the overhead of such a scheme is negligible, adding control nodes into the system implies minimal changes in the router functionalities and the total number of control nodes in the tree is not very large. Table III shows the number of control nodes required for each of tree sizes considered in the experiments.

Note that the size of the multicast tree is limited by the chosen parameter set. Since AHEC framework guarantees D_{target} and P_{target} for the application, they limit the total end-to-end delay for the worst receiver, which in our case corresponds to the depth of multicast tree. Depth of the multicast tree is a critical parameter, while the width of the tree in terms of the number of branches, could be chosen arbitrarily large in general. In our special case depth of a 100-node multicast tree, which was constructed according to the power-law degree distributions typical for the Internet-like topologies, can achieve 15 hops in some cases, and with the maximum $RTT_{link} = 50ms$ it can easily exceed the available $D_{target} = 200ms$. This makes trees with bigger depths initially infeasible. The limitation applies only to the chosen application parameter set. The whole multi-stage multicast architecture could easily be applied to the wider range of application scenarios, and we believe has a potential to provide the desired QoS for much bigger multicast client groups.

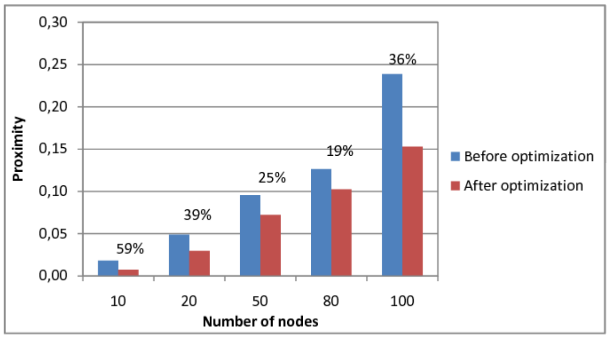

Fig. 3. The improvement observed in the proximity (estimated in percent from the value before optimization).

TABLE III
TOTAL NUMBER OF CONTROL NODES

Tree size	min	max	$average$
10	1	3	1.6
20	1	6	4.1
50	2	17	9.4
80	6	21	14.0
100	7	22	15.9

VI. CONCLUSIONS AND FUTURE WORK

A multi-stage multicast architecture was proposed to provide scalability of the multicast transmission for a wide range of multimedia Internet applications. This approach reduces the total network load in the real-time multicast scenarios with heterogeneous receivers by optimizing the amount of redundancy information required for efficient traffic protection with AHEC, keeping it close to the theoretical Shannon limit.

To further improve this work the authors will investigate a more sophisticated cost calculation that also incorporates the round-trip time (RTT), packet loss rate (PLR) and the correlation factor introduced by the statistical channel model presented in [14].

References

[1] Cisco Systems. Hyperconnectivity and the Approaching Zettabyte Era. Technical report, April 2011.

[2] D. Dolev, O. Mokryn, and Y. Shavitt. On multicast trees: structure and size estimation. *IEEE/ACM Trans. Netw.*, 14(3):557–567, 2006.

[3] M. Gorius, J. Miroll, M. Karl, and T. Herfet. Predictable reliability and packet loss domain separation for IP media delivery. In *IEEE International Conference on Communications (ICC)*, June 2011.

[4] M. Karl, M. Gorius, and T. Herfet. Routing: Why less intelligence sometimes is more clever. In *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (ISBMSB)*, 2010.

[5] M. Karl, M. Gorius, and T. Herfet. A distributed multilink media transport approach. In *International Conference on Intelligent Network and Computing*, November 2010.

[6] M. Karl and T. Herfet. On the efficient segmentation of network links. In *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (ISBMSB)*, June 2011.

[7] S. Kopparty, S. V. Krishnamurthy, M. Faloutsos, and S. K. Tripathi. Split TCP for mobile ad hoc networks. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM 2002*, pages 138–142, 2002.

[8] S. Lin and D. Costello, Jr. *Error Control Coding: Fundamentals and Applications*. Prentice Hall, Englewood Cliffs, NJ., 1983.

[9] S. Paul, K. K. Sabnani, J. C. Lin, and S. Bhattacharyya. Reliable multicast transport protocol (RMTP). *Selected Areas in Communications, IEEE Journal on*, 3:407 – 421, 1997.

[10] H. Radha and M. Wu. Overlay and peer-to-peer multimedia multicast with network-embedded FEC. In *IEEE International Conference on Image Processing (ICIP)*, 2004.

[11] Y. Shan, I. V. Bajic, S. Kalyanaraman, and J. W. Woods. Overlay multi-hop FEC scheme for video streaming over peer-to-peer networks. *IEEE ICIP*, 2004.

[12] G. Tan and T. Herfet. Optimization of an RTP level hybrid error correction scheme for DVB services over wireless home networks under strict delay constraints. *Broadcasting, IEEE Transactions*, 53:297, 2007.

[13] G. Tan and T. Herfet. Hybrid error correction schemes under strict delay constraints. In *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, 2010.

[14] G. Tan and T. Herfet. On the architecture of erasure error recovery under strict delay constraints. In *14th European Wireless Conference*, June 2008.

[15] M. Wu and H. Radha. Distributed network-embedded FEC for real-time multicast applications in multi-hop wireless networks. *Wireless Networks*, 16(5):1447–1458, 2010.