Reducing the Requirement in FEC Codes via Capillary Routing

Emin Gabrielyan
Switzernet Sàrl and EPFL
Lausanne, Switzerland
emin.gabrielyan@switzernet.com

Roger D. Hersch
École Polytechnique Fédérale de
Lausanne (EPFL), Switzerland
rd.hersch@epfl.ch

Abstract

In off-line packetized streaming, rateless Forward Error Correction (FEC) codes spectacularly improve the reliability of transmission over lossy networks. This success relies on time diversity, which in its turn relies on unrestricted buffering time at the receiver. In real-time streaming the playback buffering time is very limited (shorter than one second) and even strong FEC codes cannot protect single path communication against failures lasting longer than the buffering time at the receiver. Path diversity is a strategy that is orthogonal to time diversity and can make FEC applicable also in case of limited buffering time of real-time streaming. In this paper we introduce capillary routing algorithm offering layer by layer a wide range of multi-path routing topologies of increasing path diversity. We introduce Redundancy Overall Requirement (ROR), which for a given multi-path routing is the coefficient of the total number of redundant FEC packets to be transmitted by the sender for protecting the communication against non-simultaneous link failures. A dozen of capillary routing layers, built on several hundreds of network samples obtained from a random walk wireless Mobile Ad-Hoc Network (MANET), are rated with ROR. We show that flow diversity patterns built by capillary routing algorithm reduce substantially the amount of FEC codes required for protection of communication.

1. Introduction

Erasure resilient FEC codes achieve high reliability in off-line streaming in most challenging network conditions [Shokrollahi04], [Honda04], [Luby02], [Hollywood03]. Third Generation Partnership Project (3GPP), recently adopted Raptor [Shokrollahi04] as a mandatory code in Multimedia Broadcast/Multicast Service (MBMS), for its significant performance in file transfer.

The above examples of off-line streaming can significantly benefit from FEC due to the fact that in contrary to real-time streaming, the receiver is not obliged to deliver in time the "fresh" packets to the user and long buffering is not a concern. When buffering time is restricted, FEC can only mitigate short granular failures. Many studies reported weak or negligible improvements from applications of FEC to real-time streaming. In [Johansson02] it has been shown improvements from the application of FEC only if the stochastic packet losses range is between 1% and 5%. For real-time packetized streaming the author of [Huang05] proposed to combine FEC with retransmissions. In [Padhye00] a high overhead has been reported from the use of FEC during bursts. The author of [Altman01] claims that for delay-sensitive real-time communications, the application of FEC on the packet level can not give any valuable results at all.

In real time streaming packets representing the same information cannot be collected at very remote periods of time. However, there is an emerging body of a literature showing the applicability of FEC in real-time streaming with path diversity. Studies stressing the poor FEC efficiency assumed that the media stream follows a single path. Author of [Qu04] shows that strong FEC improves video communication following two disjoint paths and that in two correlated paths weak FEC is still advantageous. Authors of [Tawan04], [Ma03] and [Ma04] studied the path diversity in MANET. Authors of [Nguyen02] and [Byers99] studied video streaming from multiple servers. The same author [Nguyen03] later studied real-time streaming over two paths using a static Reed-Solomon RS(30,23) code (blocks carrying 23 source packets and 7 redundant packets). However the path diversity in these studies is limited to either two (possibly correlated) paths or in the best case to a sequence of parallel and serial links. Various routing topologies, so far, were not regarded as a ground for searching a FEC effective pattern.

In this paper we try to present a comparative study for various multi-path routing patterns. Single path

routing, being considered as too hostile, is excluded from our comparisons. Steadily diversifying routing patters are built by *capillary routing* algorithm where the routing suggestions are proposed layer by layer (sections 3 and 4).

In order to compare multi-path routing patterns, we introduce Redundancy Overall Requirement (ROR), a routing coefficient relying on the sender's transmission rate increases in response to individual link failures. By default, the sender is streaming the media with a static amount of FEC codes in order to tolerate a certain packet loss rate. The packet loss rate is measured at the receiver and is constantly reported back to the sender e.g. with Real-time Transport Control Protocol (RTCP). The sender increases the FEC overhead whenever the packet loss rate is about to exceed the tolerable limit. This end-to-end adaptive FEC mechanism is implemented entirely on the end nodes, at the application level, and is not aware of the underlying routing scheme [Kang05], [Xu00], [Johansson02], [Huang05] and [Padhye00]. The overall number of transmitted adaptive redundant packets for protecting the communication against link failures is proportional *(1)* to the usual packet transmission rate of the sender, *(2)* to the duration of the communication, *(3)* to the single link failure rate, *(4)* to the single link failure duration and *(5)* to the ROR coefficient of the routing followed by the communication flow. The novelty brought by ROR is that a routing topology of any complexity can be rated by a single scalar value (section 2).

In section 5, we present ROR coefficients of different routing layers built by capillary routing algorithm. Network samples are obtained from a wireless random walk Mobile Ad-hoc Network (MANET) with several hundreds of nodes. Path diversity increase achieved by capillary routing algorithm reduces substantially the amount of FEC codes required from the sender.

2. Redundancy Overall Requirement

We propose to combine the little static tolerance of the media stream, combating weak failures, with a dynamically added adaptive FEC combating the strong failures exceeding the tolerable packet loss rate.

For a given routing scheme, ROR is defined as the sum of all transmission rate overheads required from the sender for combating correspondingly all non-simultaneous link failures. For example, if the communication footprint consists of five links, and in response to each individual link failure the sender increases the packet transmission rate by 25%, then ROR will be equal to the sum of these five FEC transmission rate increases, i.e. $ROR = 5 \cdot 25\% = 1.25$. If P is the usual packet transmission rate and P_l is the increased rate of the sender, responding to the failure of a link $l \in L$, where L is the set of all links, then:

$$ROR = \sum_{l \in L} \left(\frac{P_l}{P} - 1 \right) \quad (1)$$

Let us consider a long communication, and let D be the total failure time of a single network link during the whole duration of the communication. D is the product of the average duration of a single link failure, the frequency of a single link failure and the total communication time. According to equation (1):

$$D \cdot P \cdot ROR = D \cdot P \cdot \sum_{l \in L} \left(\frac{P_l}{P} - 1 \right) \quad (2)$$

$$= \sum_{l \in L} \left(D \cdot P_l - D \cdot P \right) \quad (3)$$

Assuming a single link failure at a time and a uniform probability and duration of link failures, according to equation (3), $D \cdot P \cdot ROR$ is the number of adaptive redundant packets that the sender actually needs to transmit in order to compensate for all network failures occurring during the total communication time. Therefore ROR is a routing coefficient of the overall amount of required redundancy.

Redundant packets are injected into the original media stream for every block of M source packets using systematic erasure resilient codes. During streaming, M is supposed to stay constant. However, the number of redundant packets for each block of M media packets is variable, depending on the conditions of the erasure channel. The M source packets with their related redundant packets form a FEC block. Let us denote by FEC_p the FEC block size chosen by the sender in response to a packet loss rate p. We assume that by default the media is streamed in FEC blocks of length of FEC_t such that the flow has a static tolerance to losses $0 \le t < 1$. When the loss rate p measured at the receiver is about to exceed the tolerable limit t, the sender increases its transmission rate by injecting additional redundant packets.

The random packet loss rate, observed at the receiver during the failure time of a link in the communication path, is the portion of the traffic being still routed toward the faulty link. Thus a complete failure of a link l carrying according to the routing pattern a relative traffic load of $0 \le r(l) \le 1$ will produce at the receiver a packet loss rate equal to the same relative traffic load $r(l)$.

Equation (1) for ROR can thus be re-written as follows:

$$ROR = \sum_{l \in L \mid t \le r(l) < 1} \left(\frac{FEC_{r(l)}}{FEC_t} - 1 \right) \qquad (4)$$

a sum over all links carrying a flow exceeding the tolerable loss limit

The links carrying the entire traffic are skipped in the sum index of equation (4), since the FEC required for the compensation of failures of such links is infinite. By construction (sections 3 and 4) none of the considered multi-path routing schemes passes its entire traffic through a non-critical single link.

We compute the FEC_p function assuming a Maximum Distance Separable (MDS) code [Seroussi86], [Schwarz02]. With MDS code we can successfully decode the *M* source packets if we receive any *M* packets of the transmission FEC block.

In order to collect a mean of *M* packets at the receiver under random loss rate *p*, $M/(1-p)$ packets must be transmitted at the sender. However the probability of receiving $M-1$ packets or $M-2$ packets (which makes the decoding impossible) remains high. In order to maintain a very low probability δ of receiving less than *M* packets, we must send much more redundant packets in the block than is necessary to receive an average of *M* packets at the receiver side. We must fix the acceptable Decoding Error Rate (DER), such that $\delta \le DER$, in order to compute the $FEC_p \ge M$ function.

The probability of having exactly *n* losses (erasures) in a block of *N* packets with a random loss probability *p* is computed according to the binomial distribution:

$\binom{N}{n} \cdot p^n \cdot q^{N-n}$, where $\binom{N}{n} = \frac{N!}{n! \cdot (N-n)!}$ and $q = 1-p$

The probability of having $N-M+1$ or more losses, i.e. the decoding failure probability, is computed as follows:

$$\delta = \sum_{n=N-M+1}^{N} \binom{N}{n} \cdot p^n \cdot q^{N-n} \qquad (5)$$

Therefore for computing the carrier block's minimal length for a satisfactory communication, it is sufficient to steadily increase the block length *N* until the desired decoding error rate (DER) is met.

Transmission rate increase factors (FEC_p / M) for *M* from 1 to 10 are plotted in Fig. 1 (for $DER = 10^{-5}$). The FEC_p / M functions of Fig. 1 are compared with the lowest theoretically possible transmission rate increase factor $1/(1-p)$. The higher the number of media packets in the block the closer the transmission rate increase can approach the lowest theoretical limit.

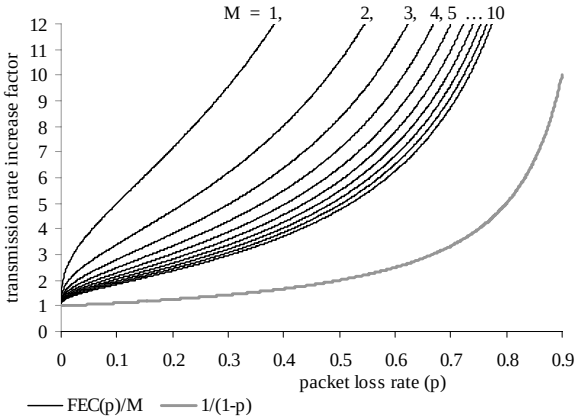

Fig. 1. Transmission rate increase factor as a function from the packet loss rate ($DER = 10^{-5}$)

By default, the playback buffer at the receiving side of the media application is designed to compensate for the network jitter and to reorder packets arriving in the wrong order. For streaming with redundant packets, the receiver must also hold in the playback buffer enough packets to restore the recoverable losses. The larger the number of media packets *M* in the FEC block, the smaller the cost of FEC overhead is, but the longer the buffering time at the receiver must be. In VOIP, for example, with a 20 ms sampling rate (of g729r8 or AMR codec) the number of media packets *M* in a single FEC block must not exceed 20 – 25 packets.

The next two sections present capillary routing construction algorithms. Section 5 presents ROR ratings of the routing layers built by capillary routing algorithm.

3. Capillary Routing

Capillary routing may be implemented by an iterative Linear Programming (LP) process transforming a single-path flow into a capillary route. First minimize the maximal value of the load of links by minimizing an upper bound value applied to all links. The full mass of the flow will be split equally across the available parallel routes. Find the bottleneck links of the first layer (see section 4). By maintaining the first upper bound (applied to all links) on its minimal level, minimize the maximal load of the remaining links by minimizing a new upper bound value applied to all links except the bottleneck links of the first layer. This second iteration discovers the sub-routes and the sub-bottlenecks of the second layer. Continue by minimizing the maximal load of the remaining links, now also without the bottlenecks of the second layer (maintaining the first and the second upper bounds at their lowest level). Repeat the iteration until the entire communication footprint of the flow is

discovered. A flow traversing a large dense network with hundreds of nodes may have hundreds of capillary routing layers.

Fig. 2, Fig. 3 and Fig. 4 show three layers of the capillary routing on a small network example. The top node on the diagrams is the sender and the bottom node is the receiver. All links are oriented from top to bottom.

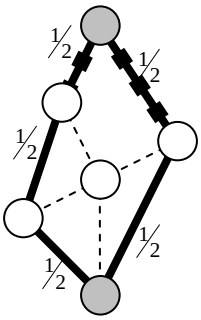

Fig. 2. In the first layer the flow is equally split across two paths, two links of which, marked by thick dashes, are the bottlenecks.

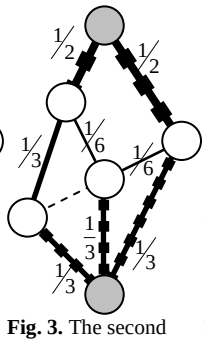

Fig. 3. The second layer minimizes to 1/3 the maximal load of the remaining seven links and identifies three bottlenecks.

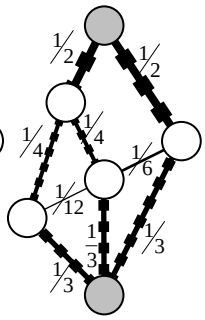

Fig. 4. The third layer minimizes to 1/4 the maximal load of the remaining four links and identifies two bottlenecks.

Although the described LP process is completely valid, it is numerically instable, the precision errors, propagating through the layers of capillary routing, reach noticeable sizes and, when dealing with tiny loads, result in infeasible LP problems. We have found a different, stable LP method maintaining the values of parameters and variables always in the same scale.

Instead of decreasing the maximal value of loads of the links, the routing path is discovered by solving max flow problems defined by the flow-out coefficients at each node. Initially only the peer nodes have non-zero flow-out coefficients: +1 for the source and –1 for the sink (Fig. 5 and Fig. 6). At each subsequent layer (Fig. 7 to Fig. 10) we have a bounded multi-source/multi-sink problem: a uniform flow from a set of sources to a set of sinks, where all rates of transmissions by sources and all rates of receptions by sinks increase proportionally in respect to each node's flow-out coefficient (either positive or negative). The multi-source/multi-sink problems arise, since the LP problem at each successive layer is obtained by complete removal of the bottlenecks from the previous LP problem, adjusting correspondingly the flow-out coefficients of the adjacent nodes (to respect the flow conservation rule) and thus possibly producing new sources and sinks in the network. Except for the unicast problem of the first layer, the successive layer problems do not belong in general to the simple class of "network linear programs" [Fourer03].

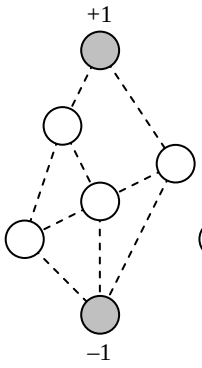

Fig. 5. Initial problem with one source and one sink node

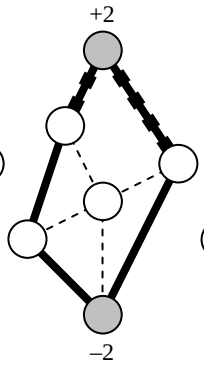

Fig. 6. Maximize the flow, fix the new flow-out coefficients at the nodes and find the bottleneck links (layer 1, $F^1 = 2$)

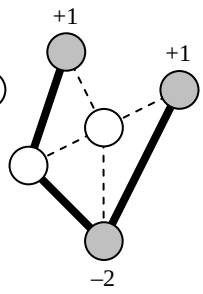

Fig. 7. Remove the bottleneck links from the network and adjust the flow-out coefficients at the adjacent nodes

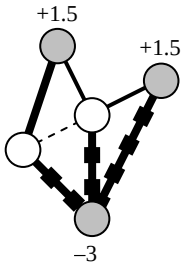

Fig. 8. Maximize the flow in the new sub-problem, fix the new flow-out coefficients at the nodes and find the new bottlenecks (layer 2, $F^2 = 1.5$)

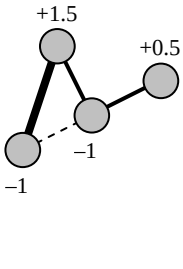

Fig. 9. Again remove the bottleneck links from the network and adjust correspondingly the flow-out coefficients at the adjacent nodes

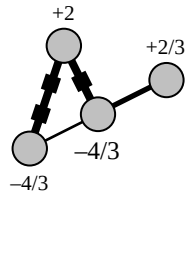

Fig. 10. Maximize the flow in the obtained new problem fixing the new resulting flow-out coefficients at the nodes and find the new bottlenecks (layer 3, $F^3 = 4/3$)

We define the bounded multi-source/multi-sink problem at layer *l* by the sets of nodes and links and by the flow-out coefficients for sources and sinks (all indexed with an upper index *l*) as follows:

- set of nodes N^l,
- set of links $(i, j) \in L^l$, where $i \in N^l$ and $j \in N^l$,
- and flow-out coefficients f_i^l for all $i \in N^l$
- at layer l the max-flow solution yields the flow increase factor F^l and the set of bottlenecks B^l, where $B^l \subset L^l$

Then, the equations for computing the sets N^{l+1}, L^{l+1} and the flow-out coefficients f^{l+1} of the next layer are as follows:

- $N^{l+1} = N^l$ (6)
- $L^{l+1} = L^l - B^l$ (7)
- $f_j^{l+1} = f_j^l \cdot F^l + \sum_{(i,j)\in B^l}(+1) + \sum_{(j,k)\in B^l}(-1)$ (8)

add 1 for each incoming bottleneck link (i, j)

subtract 1 for each outgoing bottleneck link (j, k)

After a certain number of applications of the max-flow objective with corresponding modifications of the problem, we will finally obtain a network having no source and sink nodes. At this moment the iteration stops. All links followed by the flow in the capillary routing are enclosed in bottlenecks of one of the layers.

In order to restore the original proportions of the flow, the flow increases, induced by the preceding max-flow solutions must all be compensated. The true value of flow $r_{i,j}$ traversing the bottleneck link $(i,j) \in B^l$ of layer l is the initial single unit of flow divided by the product of the flow increase factors F^i (where $1 \le i \le l$) of the present and all preceding layers:

$$r_{i,j} = \frac{1}{\prod_{i=1}^{l} F^i} \quad \begin{array}{l}\text{where } l \text{ is the} \\ \text{layer for which} \\ (i,j) \in B^l\end{array} \qquad (9)$$

The max-flow approach proves to be very stable, because it maintains all values of variables and parameters within a close range of unity (even for very deep layers with tiny loads) and also because it enables to validate and if necessary re-calibrate all possible errors in the flow-out coefficients of the LP problem formulated for the next layer of capillary routing.

In the next section we show how to identify bottlenecks after the max-flow solution of the capillary routing layer is found.

4. Bottleneck hunting loop

Bottlenecks of each max-flow solution are discovered in a bottleneck hunting loop. Each iteration of the hunting loop is an LP cost minimizing problem that reduces the load of the traffic over all links having maximal load and being suspected as bottlenecks. Only links maintaining their load at the initial maximal level will be passed to the next iteration of the hunting loop. Links whose load has been reduced under the LP objective are not bottlenecks and removed from the list of candidates. The bottleneck hunting iteration stops if there are no more links to remove.

Let us show the bottleneck hunting on the example of Fig. 11 with three transmitting nodes and two receiving nodes. The flow can be proportionally increased at most by a factor of 4/3, such that each flow-out coefficient at sources become equal to 4/3 and each flow-in coefficient at sinks become equal to -2 (see Fig. 12). The bottleneck links are among four maximally loaded suspected links {*a*, *b*, *d*, *e*}, marked in Fig. 12 by thick dashes.

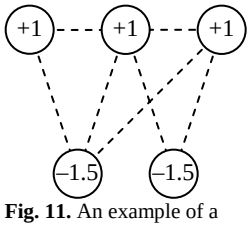

Fig. 11. An example of a bounded multi-source/multi-sink problem (obtained during construction of the capillary routing from a network with one source and one destination nodes)

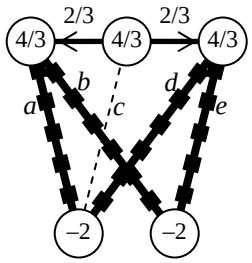

Fig. 12. A max-flow solution with the flow increase factor of 4/3, containing four maximally loaded candidate links {*a*, *b*, *d*, *e*}

An LP cost-minimizing objective can reduce the sum of loads of all four suspected links from the initial value of 4 (see Fig. 12) to the minimal value of 3 (see Fig. 13). In this min-cost solution previously suspected link *d* does not maintain anymore its load on the maximum and thus there left only three suspected links {*a*, *b*, *e*}, marked in Fig. 13 by thick dashes.

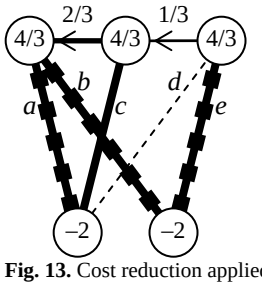

Fig. 13. Cost reduction applied to four fully loaded links of Fig. 12 reduces the load of suspected link *d*, and the suspect list is now {*a*, *b*, *e*}.

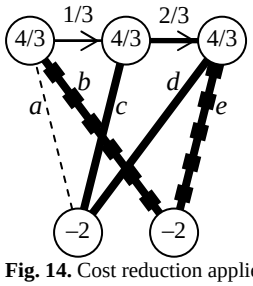

Fig. 14. Cost reduction applied to the three fully loaded links of Fig. 13 reduces the load of another suspected link *a*, and the true bottleneck links are {*b*, *e*}.

The sum of loads of now three suspected links can be further reduced from the value of 3 (see Fig. 13) to a minimal value of 2 (see Fig. 14). Now there are only two links {*b*, *e*} maintaining their loads at the maximal value, marked in Fig. 14 by thick dashes. These two remaining links are the true bottleneck links, since additional applications of the LP cost minimizing objective does not result in any further reduction of the sum of loads.

In this example the two bottlenecks were found in two iterations; for larger networks with hundreds of nodes the bottleneck hunting of each capillary routing layer can take dozens of iterations.

For capillary routing patterns, built simultaneously on 200 unbounded networks samples with 300 nodes in each (total 60,000 nodes and 511,140 links), Fig. 15 shows the decrease of the number of suspected links during the bottleneck hunting loop of each capillary routing layer. Depending on the layer of the capillary routing (1 to 10), the bottleneck hunting loop takes 14 to 23 iterations.

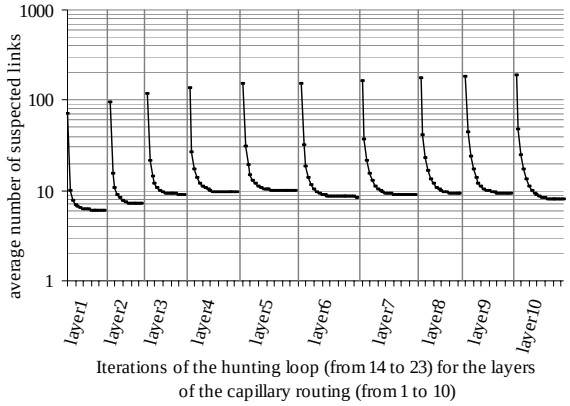

Fig. 15. Decrease of the number of suspected links during the bottleneck hunting loop of each of 10 capillary routing layers

Each iteration of the bottleneck hunting loop removes from the suspect list numerous non-bottleneck links. At the end of each hunting loop the suspect list consists of only true bottleneck links. The average number of true bottleneck links at each layer is between 5.9 and 9.9.

5. FEC requirement in capillary routing

We compute the average ROR coefficient simultaneously over several networks for capillary routing layers 1 to 10. In Fig. 16, we considered eight different sets of samples, each containing 25 distinct networks. At the same time we consider also media streaming at different default intensities of the static FEC codes tolerating respectively packet loss rates from 3.3% to 7.5%. For each set of samples and for each static FEC intensity we show how the average ROR coefficient changes as the capillary routing layer increases.

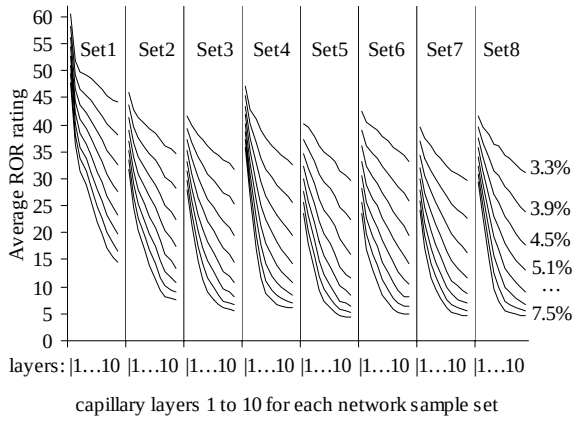

Fig. 16. Average ROR as a function of the capillary routing layer (the static tolerance of the stream from 3.3%, for the upper curves, to 7.5%, for the lower curves, by a step of 0.6%)

The drawback of path diversity is that long paths can be formed increasing unjustifiably the number of links in the path, consecutively the overall failure rate, and finally the overall requirement in FEC codes. Our measurements show that despite the communication footprint becomes larger, with the routing patters built by capillary routing algorithm, the requirement in redundant packets decreases noticeably most of the time.

Logically, the ROR curve of the media stream is shifted down as the statically added tolerance increases. At the same time the presence of a higher static tolerance yields a much stronger efficiency gain achieved by the deeper routing layers.

Although there are hundreds layers in the complete capillary routing, the first few layers alone reduce the average FEC effort of the sender by a factor of four. According to the chart, streams tolerating a 6% or higher packet loss rate almost do not gain from spreading beyond layer 8.

The exact pattern of the ROR improvement curve, as a function of the layer, depends on the distance between the peers, the network size and its density. The network samples for the above chart are drawn from a random walk wireless Mobile Ad-hoc Network (MANET). Initially the nodes are randomly distributed on a rectangular area, and then, at every timeframe, they move according to a random walk algorithm. If two nodes are close enough (and are within the coverage range) then there is a link between them. In the above example, there are 300 nodes and 200 time-frames, each leading to a separate network sample (all of which are distributed into eight sets represented on the above chart).

The ROR rating of routing samples is computed by equation (4), where the FEC block size (as function of the packet loss rate p) is computed according equation (5). The number of media packets (M) per transmission block is 20 and the desired decoding failure rate (DER) is 10^{-5}.

6. Path diversity for off-line streaming

In off-line applications the number of source packets M per transmission block can be very large. In file transfers for example, M can be the number of all packets in the entire file, or in one-way video, the buffering time can be a few minutes long, with thousands of media packets within each single FEC block. When the number of packets in the transmission block is very large, for a given probability p of packet losses, the proportion of actually received packets remains very close to $1-p$. Although for very large numbers of source packets MDS codes do not exist,

several other capacity approaching codes, such as erasure resilient fountain codes [MacKay05] can practically reach the theoretical limit. A packet loss rate p can be thus compensated by an increase of the encoded stream transmission rate by the lowest possible theoretical factor of $1/(1-p)$.

Path diversity can be required in off-line streaming applications or in long downloads for avoiding the idle times of the last kilometer bottleneck occurring during arbitrary failures within the lossy Internet. Thanks to the sender's adaptive transmission rate and to multi-path routing, one may feed the last kilometer bottleneck link constantly at its maximal bandwidth (see [Nguyen02] and [Byers99] for video streaming from multiple servers). Since the buffering is not a concern the application may use a capacity approaching fountain code, such that each individual network failure causing a packet loss rate p can be compensated by an increase of the transmission rate only by a theoretical factor of $1/(1-p)$. It depends now on the choice of multi-path routing pattern how much the overall amount of FEC codes required from the sender will be. The ROR coefficient of the routing pattern can be computed with the following equation derived from equation (4):

$$ROR = \sum_{l \in L \mid t \le r(l) < 1} \left(\frac{1-t}{1-r(l)} - 1 \right) \qquad (10)$$

For the same set of 200 network samples presented in section 5, we computed the average ROR coefficients now according to equation (10). As in Fig. 16 the network samples are distributed into eight sets. Off-line streaming ROR coefficients, for various intensities of static tolerance, as functions from capillary routing layers (1 to 10) are presented in Fig. 17.

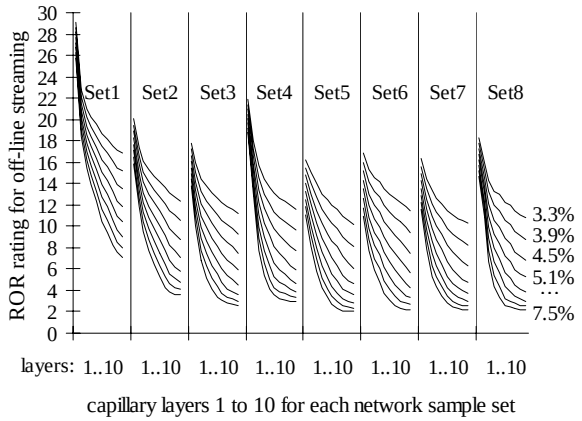

Fig. 17. Average off-line streaming ROR as a function from the capillary routing layer

In off-line streaming (Fig. 17) the Redundancy Overall Requirement is twice as low as in real-time streaming (Fig. 16), but the capillarization of routing is beneficiary in both cases.

7. Applications of capillary routing

Multi-path routing suggestions for fault-tolerant streaming are applicable not only to ad-hoc or sensor networks, but also to mobile networks, where wireless content can be streamed to and from the user via multiple base stations; or to the public internet, where, if the physical routing cannot be accessed, path diversity can be still obtained at the application level.

Spreading in IP networks (without having access to the physical routing) can be achieved transparently by splitting the flow at the source across VPN tunnel interfaces leading to several VPN gateways scattered across the network. Alternatively, path diversity can be also obtained by assigning to media gateways several IP interfaces connected to networks of different Internet Service Providers (ISP). A more flexible method relies on overlay networks over the public Internet using peer-to-peer relay nodes (see [Nguyen03] and [Guven04]).

Modifying the physical routing by the ISP requires the least overhead. Most of the UDP packets carry streaming media, and since capillary routing is good for any type of real-time or off-line streaming media, an ISP can simply spread out the routing of the entire mix of UDP packets traversing its network. The ISP needs to properly balance at each router the outgoing traffic across its outgoing interfaces. Most IP routers, provide load balancing in static routing mode or in Enhanced Interior Gateway Routing Protocol (EIGRP) mode (which must be used in the packet load balance mode instead of the typical session load balance mode).

8. Conclusions

The quality and reliability issues concerning real-time streaming over packet networks are of growing importance. Commercial real-time streaming applications however do not consider channel coding as a serious solution for improving the reliability of communication. In single path communications, even heavy FEC overheads cannot protect against failures lasting more than the short duration of the playback buffer. Recent studies demonstrated that path diversity makes FEC applicable for real-time streaming. By studying a wide range of routing topologies, we show that the proper choice of the routing pattern can make FEC extremely efficient. Combination of channel

coding with appropriate multi-path routing improves the reliability of real-time packetized communications even in the case of very short playback buffers.

We introduce a layer by layer strategy for building multi-path capillary routings. The first layer provides a simple max-flow solution. As the layer number increases, the spreading of the underlying routing scheme makes the network more secure for real-time streaming. For a given source and destination there exists only one solution of capillary routing and it does not depend on the particular characteristics of the streaming (buffer size, coding method, etc).

We introduce ROR, a method for rating multi-path routing patterns by a single scalar value. The ROR rating corresponds to the total redundancy overhead that the sending node must provide in order to combat the losses occurring from non-simultaneous failures of links in the communication path.

Despite the fact that spreading out of the routing results in the increase of the failure rate of underlying links and consecutively also of the transmissions of adaptive FEC codes; however, by using routing patterns built by capillary routing algorithm, the overall amount of FEC codes decreases substantially.

9. References

[Shokrollahi04] Amin Shokrollahi, "Raptor codes", ISIT'04, June 27 – July 2, page 36

[Honda04] Loring Wirbel, "Deal pushes algorithms into digital radio", April 13, 2004, http://www.commsdesign.com/showArticle.jhtml?articleID=18901216

[Luby02] Michael Luby, "LT codes", FOCS'02, November 16-19, pp. 271-280

[Hollywood03] Mark Fritz, "Digital Dailies Flow Freely from Fountain", April 1, 2003, http://www.emedialive.com/Articles/ReadArticle.aspx?CategoryID=45&ArticleID=5077

[Johansson02] Ingemar Johansson, Tomas Frankkila, Per Synnergren, "Bandwidth efficient AMR operation for VoIP", Speech Coding 2002, Oct 6-9, pp. 150-152

[Huang05] Yicheng Huang, Jari Korhonen, Ye Wang, "Optimization of Source and Channel Coding for Voice Over IP", ICME'05, Jul 06, pp. 173-176

[Padhye00] Chinmay Padhye, Kenneth J. Christensen, Wilfrido Moreno, "A new adaptive FEC loss control algorithm for voice over IP applications", IPCCC'00, Feb 20-22, pp. 307-313

[Altman01] Eitan Altman, Chadi Barakat, Victor M. Ramos, "Queueing analysis of simple FEC schemes for IP telephony", INFOCOM 2001, Vol. 2, Ap 22-26, pp. 796-804

[Qu04] Qi Qu, Ivan V. Bajic, Xusheng Tian, James W. Modestino, "On the effects of path correlation in multi-path video communications using FEC over lossy packet networks", IEEE GLOBECOM'04 Vol. 2, Nov 29 - Dec 3, pp. 977-981

[Tawan04] Tawan Thongpook, "Load balancing of adaptive zone routing in ad hoc networks", TENCON 2004, Vol. B, Nov 21-24, pp. 672-675

[Ma03] Rui Ma, Jacek Ilow, "Reliable multipath routing with fixed delays in MANET using regenerating nodes", LCN'03, Oct 20-24, pp. 719-725

[Ma04] Rui Ma, Jacek Ilow, "Regenerating nodes for real-time transmissions in multi-hop wireless networks", LCN'04, Nov 16-18, pp. 378-384

[Nguyen02] Thinh Nguyen, Avideh Zakhor, "Protocols for distributed video streaming", Image Processing 2002, Vol. 3, Jun 24-28, pp. 185-188

[Byers99] John W. Byers, Michael Luby, Michale Mitzenmacher, "Accessing multiple mirror sites in parallel: using Tornado codes to speed up downloads", INFOCOM 1999, Vol. 1, Mar 21-25, pp. 275-283

[Nguyen03] Thinh Nguyen, P. Mehra, Avideh Zakhor, "Path diversity and bandwidth allocation for multimedia streaming", ICME'03 Vol. 1, Jul 6-9, pp. 663-672

[Kang05] Seong-ryong Kang, Dmitri Loguinov, "Impact of FEC overhead on scalable video streaming", NOSSDAV'05, Jun 12-14, pp. 123-128

[Xu00] Youshi Xu, Tingting Zhang, "An adaptive redundancy technique for wireless indoor multicasting", ISCC 2000, Jul 3-6, pp. 607-614

[Seroussi86] Gadiel Seroussi, Ron M. Roth, On MDS extensions of generalized Reed- Solomon codes, IEEE Transactions on Information Theory, Vol. 32, Issue 3, May 1986, pp. 349-354

[Schwarz02] Thomas S. J. Schwarz, Generalized Reed Solomon codes for erasure correction in SDDS, In Workshop on Distributed Data and Structures, WDAS 2002, Paris, Mar 2002

[Fourer03] Robert Fourer, *A modeling language for mathematical programming*, Thomson – Brooks/Cole, second edition, 2003, page 343

[MacKay05] David J. C. MacKay, "Fountain codes", IEE Communications, Vol. 152 Issue 6, Dec 2005, pp. 1062-1068

[Guven04] Tuna Guven, Chris Kommareddy, Richard J. La, Mark A. Shayman, Bobby Bhattacharjee "Measurement based optimal multi-path routing", INFOCOM 2004, Vol. 1, Mar 7-11, pp. 187-196