Capillary multi-path routing with Forward Error Correction

Emin Gabrielyan
Switzernet – EPFL
1015 Lausanne – Switzerland
emin.gabrielyan@{switzernet.com,epfl.ch}

HTML – PDF – DOC

Table of contents:

I. Introduction

Media streaming is becoming increasingly important in Internet. The major cause for multimedia quality degradation in IP-networks is packet loss. Packet loss occurs in network due to bursts and link failures or packets may be dropped in the application, if they are received too late.

Forward Error Correction is a mechanism that may allow the streaming application to overcome losses without utilizing retransmissions. Real-time media or files sent over the internet are chopped into packets, and each packet is either received without error or not received. Packetized communications over internet thus behave like erasure channels and the erasure resilient FEC codes, such as Reed Solomon or another Maximum Distance Separable (MDS) code can be applied on the packet level to a stream of UDP (User Datagram Protocol) data packets.

With the proper amount of redundant data included in the stream of transmitted packets, erasure resilient FEC can mitigate the impact of packet loss on the data. For numerous packetized applications, employment of erasure resilient FEC codes offered spectacular

results: in satellite feedback-less broadcasts of recurrent voluminous updates of GPS (Global Positioning System) maps to millions of motor vehicles under the condition of arbitrary fragmental visibility due to their locations and trajectories: tunnels, underground parking, buildings and garages [honda04]; in a film industry for a fast delivery over Internet of the day's film footage from the location it has been shot to the studio that is many thousand miles away [hollywood03] using LT codes [Luby02]. 3GPP, 3rd Generation Partnership Project, recently adopted Raptor [Shokrollahi04] as a mandatory code in Multimedia Broadcast/Multicast Service (MBMS), a part of 3G standard, for its low CPU usage and for the significant performance in media broadcasting and file transfer [MBMS05a], [MBMS05b], [MBMS05c].

The above references are rather examples of off-line applications, where there are no constrains on the buffering time because the sender is not obliged to deliver in time the "fresh" packets of a very short life time.

Regarding from the stand point of a real-time application, in the film transfer or in the GPS map update, the buffering time is practically unlimited. Several publications studied application of FEC in real-time media streaming. [Johansson02] reported improvements from the application of FEC if the packet losses range is between 1% and 5%. [Huang05] proposed to combine retransmissions, if the round trip time (RTT) is low, with FEC, and reported improved speech quality perception in a practical experiment. [Padhye00] reported high overheads from the use of FEC during bursts but proposed to survey the history of losses (experienced by the application) after reconstruction to choose a good FEC. Other publications have shown that for two-way, delay-sensitive real-time communications, the application of FEC on the packet level can not give any valuable results at all [Altman01].

In the off-line streaming success examples (MBMS or GPS map update) it was the long (or almost unlimited) buffering time that let FEC be an efficient solution of the fault tolerance. However according to [Altman01] in real-time applications with limited buffering time, only negligible improvement from application of FEC can be obtained at the cost of very high overheads. Voice over IP, a widely demanded interactive application, is an example of a poor FEC efficiency: the ratio of the lossy network path's failure time over the permitted limit on the buffering time is too high.

If it is still believed that FEC can however significantly improve the tolerance of real-time streaming, we must exploit other dimensions which can "replace" the long buffering time and open to FEC the true perspectives in the domain of real-time streaming.

All previous studies stressing on the poor FEC efficiency assumed that the media stream always follows a single path. Short granular failures can be still mitigated at a certain cost of the FEC overhead, but obviously, there is no FEC which can solve a problem of a complete link failure on the single route path if the failure duration is long enough to overpass the permitted maximum of the buffering time (long bursts or congestions, hardware or software circuit outages until the recovery triggering, deny of service – DOS attacks).

Application of FEC to the real-time streaming however is not hopeless, because, there exists the other unexploited dimension: the underlying network routing, an alternative orthogonal method toward the buffering. The previous studies ignored the effort that could have been provided by the underlying routing.

The advantage of the flow following multiple paths versus a single path routing is obvious: a long link failure on the single path route leaves the application completely helpless, but the application can apply FEC to recover a single link failure of a spread multi-path route.

There is an emerging body of a literature addressing the advantages of multi-path routing in media streaming. The author of [Qu04] studied video communication following alternate paths and has shown that in disjoint paths the advantages of strong FEC are significant; in the correlated paths weak FEC can be still advantageous. In [Tawan04] the author proposed path diversity in Ad-Hoc network motivated by the load balance and capacity increase concerns but mostly by the choice of the adaptive routing protocol; the achieved efficiency in use of FEC was also mentioned. In [Ma03] and [Ma04] it was suggested to replace in MANET the link level Automatic Repeat Query (ARQ) by a link level FEC assuming intelligent regenerating nodes. The routing topology is simplified into a path with serial or parallel links between the nodes and it was suggested that the end-to-end packet loss rate as a function from hops will improve. However in such a hop by hop based FEC, since the retransmission of a lost media packet may be delayed until its recovery, for the stability of the real-time media the node needs a jitter buffer (of the size of the transmission block) and the jitter buffer's constant delay will be propagated therefore with each hop, weakening this approach for the real-time streaming. In [Nguyen02] it was proposed the server diversity in one-way video, demonstrating that streaming to a single receiver from multiple servers improves the tolerance and increases the capacity. The author is focusing on the receiver driven protocol for allocating to sending servers the rates (based on RTT values measured from the receiver) and for partitioning non-overlappingly the packets to be transmitted. But for the buffered multi-point video delivery the author should have chosen instead of Reed-Solomon (RS) a rateless uniform fountain code (random linear fountain, LT or Raptor [MacKay05]) and would therefore skip from the protocol the packet partitioning part [Byers99]. The author also ignores the network topology, assuming that the routes from the senders to the client do not share links. The same author lately studied the path diversity effect, this time, on the protection of point to point real-time video streaming using a permanent RS(30,23) encoding (23 media packets and 7 redundant packets) at the sender [Nguyen03]. The author, similarly to [Qu04] compared two-path scenarios (shortest path together with an alternate completely disjoint or with an alternate but correlated path) with the single OSPF routing strategy and has shown clear advantages of the double path routing. The choice of RS versus fountain code is justified in [Nguyen03] by the limits of the real-time application on the buffering.

If one of the axes amplifying the effect of FEC was the buffering time, then the second orthogonal ax therefore is clearly the routing. While the first ax, the buffering

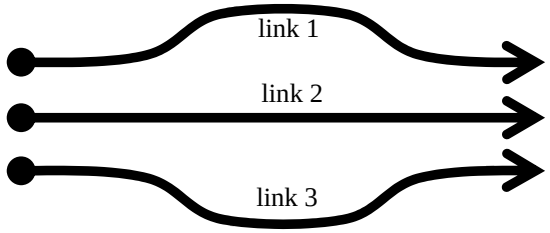
[Weatherspoon02], [Krunz04], gave excellent results in the off-line streaming, the second, the routing ax, is hopefully suitable for the protection of real-time two-way streaming, insurmountable by the buffering ax alone, due to interactiveness and delay sensitivity.

All above referred publications suggested the important and promising impact the routing can have to the efficiency of FEC. However the considered topologies were simple and limited to only two, possibly correlated, paths [Qu04] and [Nguyen03], or in the best case to a sequence of parallel and serial links [Ma03]. We have not found a study of an impact of a wider range of routing patterns to the FEC efficiency. There is lack of a work in the literature studying the routing as a space of possibilities or as a dimension, and seeking in there the optimal point of FEC efficiency – the friendliest routing pattern. From this point of view we can say that the previous studies only proved the advantageousness of path diversity, most of the time of double-paths, toward only single path routing. The author of [Nguyen03] concludes that deployment of the second redundant path along with the default path effectively results in a factor of 3 reduction in irrecoverable packet loss as compared to uni-path scheme. In this paper we try to present a global comparative study across much larger number of friendly routing patterns, all multi-path, virtually erected along a routing ax. Single path routing, being considered as too hostile, will be even excluded from our comparison system. We begin from the strategies similar to multi-path approaches of [Qu04], [Nguyen03] and [Ma03] as our starting point, but we will develop them step by step into more elaborated, dispersed and advantageous routing samples. Addressing the parallelism and spreading aspects in the routing patters we assume several abstractions and simplifications for other network model parameters (e.g. uniform links and lack of delay prorogation). To compare one spread routing with another one, we will introduce a measure of the routing's advantageousness. We will demonstrate that the first level of diversity obtained by converting a single path routing to the basic multi-path routing is not the terminal achievement of the path diversity approach. The routing can be still sensibly improved (toward FEC), layer by layer. Routing ax consisting of only spread routing patterns; can further burst the efficiency of FEC. We hope that our investigations will provide guidelines for future design and development of multi-path routing strategies in multimedia transmission.

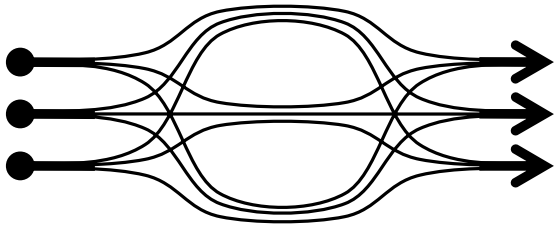
II. Path Diversity Spread Routing

Addressing the problem of fault-tolerance from the routing point of view, we propose to route a media stream in a way which minimizes the impact of individual link failures to the quality of the media. Even if failure recovery mechanisms are assumed in the network, the way we route the end to end transmission over the network must maximally protect the media before the network detects and recovers the failure. Suggested routing does not require additional capacities from the network; it simply spreads the packetized transmission over as many independent paths as possible.

If a network can sustain a given number of independent transmissions each following a single path, then in principle, the same transmissions can traverse the network in a spread fashion, without requiring additional capacity.



Transmissions following a single path


Transmissions spread over multiple paths do not require extra network capacity

Spread routing alone does not solve the problem of tolerance. If media stream is not capable to sustain any losses at all, by spreading the transmission the media becomes more vulnerable to link failures, since there are more links whose failure will damage the stream. However if media is equipped with a certain level of tolerance toward losses, the spreading of flow may protect the transmission. With sufficient redundancy and spreading factor the transmissions can be protected completely against any single link failure.

A tolerance of media to losses can be the part of the nature of the media itself. If the media is destined for a human perception (VOIP, image or video), it can still conserve its content with a quite high level of stochastic losses. With g729r8, g723r53, g723r63 and gsmfr VOIP compression codecs the speech remains comprehensible with 8% to 11% losses. Similarly, a video compression codec (e.g. MPEG) also ensure a certain tolerance to losses, e.g. due to passive error concealment (level of tolerance of video streams is lower compared with voice).

Natural tolerance of media may be not sufficient to make a real benefice from the spread routing. For any type of real-time media transmission, tolerance to losses can be obtained by injection at the source of some redundancy provided by erasure resilient forward error correction codes (e.g. Reed Solomon codes).

For the two figures above, assuming a tolerance of 1/3 for each media stream, in the single path scenario, any single link failure would kill one of the media streams. At the same time all multiple path streams will sustain a complete failure of any of three links.

Most internet routers employ First-In-First-Out (FIFO) policy in which, the successive arrived packets are dropped if the network buffer becomes full. Hence a model of

fragmental link failures can be adopted to approximate link outages, bursts and congestions. Link is either in the operational or in a faulty state and is characterized by its average failure and operation time.

It is not however very efficient to permanently transmit with the media stream a large amount of constant redundancy. It is much better to increase the redundancy on demand. The receiving node can detect the deficit of delivered packets and demand an appropriate increase of the source coding. In this paper we propose a combination of the following three factors: (1) an optimal friendly spread routing, (2) a little permanent constant tolerance to combat bursts and (3) an added layer of a dynamic adaptive end-to-end coding to combat outages, congestions and long failures.

III. End-to-End adaptive FEC

As a reference for the real-time streaming application we take the most widely used example: voice over IP. The driving motivation for transporting real-time voice over IP networks (apart the uniformity, fluidity and manageability aspects) is the significant cost savings achievable by eliminating or bypassing the circuit-switched TDM telephony infrastructure. Although use of VOIP is currently weak, the wide penetration of the packetized voice with possibly complete replacement of TDM technologies is practically inevitable.

Most frequent failures, congestions caused by the bursty traffic, are not supposed to be considered as link outages, since most of routers maintain QoS and the tiny VOIP stream must not suffer from the HTTP bursts, which are often less important from the user point of view, but may consume 25 to 50 times more bandwidth. The IP telephony service provider (ITSP) however often is different from the internet service provider (ISP, usually the former monopolist of the ex-ante regime providing internet via ADSL employing the last kilometer infrastructures) and the packets of the ITSP most of the time do not have particular priority in the network of the ISP. The bandwidth consumed by IP Telephony VOIP stream (using the high complexity codecs) with all IP, UDP and RTP header overheads is as low as 16 to 20 Kbps (the pure voice payload bandwidth is as low as 5.3, 6.3 or 8 kbps, the recently standardized AMR has 4.75 kbps mode). The average ADSL capacity is of 600 Kbps to 1 Mbps for the downstream and the forth of it for the upstream. The end user will have therefore absolutely no objection if, due to a short failure somewhere in internet, the IP telephony voice call application will be capable to temporarily burst its transmission rate several times saving thus the voice call from dropping. [Huang05] presented an implementation doubling the media packets at high losses and reported good results in practice. With a spread multi-path routing the need of a bit rate increase would be much fewer. As a reaction to failures arbitrarily occurring at different places in the network, the ability of a voice software or hardware (e.g. of a SIP phone) to periodically increase its transmission rate, maintaining the speech quality constantly at the high level during the phone conversation, will be highly appreciated and the bandwidth is not a concern. In the present study we therefore assume that all links have a sufficiently large capacity compared with the bandwidth of a media stream.

The cooperativeness or friendliness of a routing toward FEC is simpler of all to measure based on the satisfaction level of a realistic application employing end-to-end adaptive FEC. In the spread routing, at the node which has multiple outgoing links, the router randomly allocates the packets of the incoming queue to the outgoing links (according the load distribution pattern of the routing). Losses at the receiver, occurring due to temporary congestions or failures of network links have therefore a random pattern. All links of our model have an equal probability, frequency and duration of failure. Before the triggering of any failure detection and recovery mechanism, a failure of a link laying on the communication path, produces, according the portion of the traffic being still routed toward the faulty route, random losses, observed finally at the receiver. The media stream of the application is equipped with a constant tolerance t to certain level of losses ($0 \leq t < 1$). If the losses measured at the receiver are exceeding the critical level the receiver demands the sender via a feed-back channel injecting into network an extra redundancy. Either the feed-back loop is assumed negligibly short compared with the duration of a failure of a link or the usual constant tolerance of the media to random losses is high enough to leave a sufficient time to the feedback loop, without a risk of too many damages to the usability of the media stream. The sender must add to its further transmission a sufficient quantity of redundancy to compensate the new losses signaled by the receiver, thus continuing to maintain the communication constantly at the required level of quality. The sender must compute the redundancy need as a function from the percentage of losses p observed by the receiver in the stream of incoming packets. Such an end-to-end feedback based adaptive FEC mechanism is not aware of the routing and is implemented entirely on the end nodes in the application level.

Implementation of this method in practice was proposed by several authors: [Kang05] proposed to adjust the amount of FEC based on packet loss information under dynamically changing network environment, [Xu00] proposed to dynamically combine FEC with automatic repeat request. [Padhye00] proposed to slowly tune the amount of FEC considering the history of losses measured after decoding. [Johansson02] and [Huang05] proposed to apply to packet switched network the mode adaptation scheme of Adaptive Multi-Rate (AMR) codec, adopted as the standard speech codec by 3GPP. If the network is lossy, source coding is reduced and channel coding is increased and, inversely, if the streaming is lossless the source coding is increased and a weak FEC is used.

FEC can be added to the media stream in one of the following three ways. In media specific FEC methods, the redundant information is generated using a different lower bit rate encoding algorithm than the algorithm used to encode the original packet (e.g. using the eight modes of AMR). Media specific schemes piggyback information about the present period with later packets [Padhye00], [Altman01], [Johansson02]. In a similar method, but without multimode media source encoder, i.e. abstracting from the media, packets can be extended to carry one or more duplicates of previously produced samples or their XOR products (or another form of their redundancy). While the two above methods are extending the packets of the stream in size, according the third method the redundant packets (representing a number of original packets) are extending the stream in quantity, inserting the redundant packets in the stream [Huang05], [Nguyen03]. In our

model we use the third method. The following arguments are in favor of injecting the redundant packets into the stream. By adding packets of identical size we do not (or almost do not) influence on the packet loss rate (which can be a function from the link BER and the packet size or from the queue policy of routers), and thus we do not need to model individual links and routers to adjust this change. A practical advantage of adding the redundant packets without modifying the stream of original packets (using a systematic erasure resilient FEC code) is the possibility of a full transparency on the media application layer. The redundant packets can be generated and decoded back into media packets (or dropped if no media packet of the block is lost) on the edge routers (usually the NAT router of the user), leaving this process completely transparent for the media application.

Although the proposed adaptive FEC application is not aware of the underlying network topology and the routing, we define the aggregate effort of adaptive redundancy demanded from the sender by the receiver during the whole communication time as an evaluation measure of the friendliness and the advantageousness of the underlying network routing toward the tolerance. Fewer is the overall amount of the encoder's redundancy effort demanded by the receiver to the sender, higher is the rating of the routing toward the tolerance of the communication (assuming that the routing suggestions are measured under the identical pattern of network link failures). The so defined encoder's aggregate effort, called henceforth Adaptive Redundancy Overall Need (*ARON*), is the easily measurable scalar value characterizing the friendliness of the entire underlying routing (toward FEC). Any collection of routing suggestions can be sorted by their advantageousness toward tolerance, simply by measuring and comparing their ARON values.

IV. Adaptive Redundancy Overall Need (ARON)

Strictly defined ARON is the sum of all FEC overheads demanded from the sender for the compensation of sequential losses of all sub-flows carried by each individual link in the footprint of the communication path. The critical links of the path carrying the entire traffic are ignored, since the FEC required for the compensation of a failure of such links would be infinite. We assume that all routing schemes that are subject of comparisons are smart enough to delegate the load from a critical route over other links, if of course the network topology makes it possible. Thus our comparisons cannot contain a really bad routing sample which is following a single path route on a link when alternate multiple path routes are possible, and if the link is really critical by the network topology, then without a risk of affecting the comparison results such a link can be removed from all suggested routings in order to compare their remaining portions. We do not therefore study the advantage of the multiple path strategies versus single path routing, as in [Qu04], [Ma04], [Nguyen03], but we rather measure the advantageousness of routing within the scope of multi-path approaches.

If the failure periodicity is the same for all links such that any single link fails approximately every *s* seconds, if the observation duration (or the duration of the

communication) is *T* and the duration of a single failure is *d*, then $d \cdot \frac{T}{s}$ is the total failure time of any single link during the observation period.

Let *M* be the initial number of original media packets to be delivered to the receiver by the transmission blocks. Let $FEC_p \geq M$ be the needed size (number of packets) of the transmission block, under $0 \leq p < 1$ percent of random losses in the network (observed at the receiver) for maintaining the required QoS. Let *L* be the set of links laying on the communication path, and let $r(l)$ be the load of a link $l \in L$ under the present routing scheme. Let the percentage $0 \leq t < 1$ of recoverable losses, be the level of the tolerance permanently maintained in the media stream (even if there are no observed losses, good for bursty loss environment). The equation for ARON then looks as follows:

$$ARON = \sum_{l \in L \mid t \leq r(l) < 1} \frac{FEC_{r(l)} - FEC_t}{FEC_t}$$

If we know the ARON of the given routing, we can deduce the number of redundant packets that the sender will be obliged to transmit in order to compensate all arbitrary network failures during the call duration *T*. We need also to know the periodicity and the duration of an individual link failure and the block transmission rate *B* (the block transmission rate is supposed to stay constant, while the packet transmission rate may periodically rise due to temporary increase of the number of redundant packets in the block). The number of redundant packets (the formula below) gives a practical meaning to the routing parameter ARON, but we assumed here a single link failure at a time [] (e.g. short link down-time and long link up-time).

$$d \cdot \frac{T}{s} \cdot B \cdot FEC_t \cdot ARON$$

FEC_p as a function from the network's random loss probability *p* may vary depending on the type of the encoder, the number of the media packets in the block (the buffering time) and the requirement for QoS.

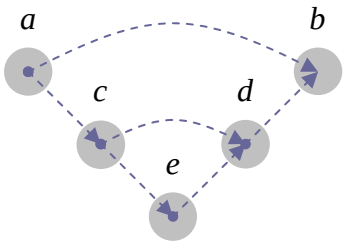
We can imagine a pseudo "real-time" file transmission application in which the reception must be maintained at a constant rate and the file transfer must be accomplished within the time of the file size over the reception rate. Thus to ensure the reception at the maximal downlink rate through a lossy network the sender must transmit at a variable rate compensating thus the arbitrary losses arising in different network locations. This could be needed for synchronous updates of multiple stations of limited downlink bandwidth or a common problem of a last kilometer bottleneck for the internet user watching a one way video [Nguyen02].

Files are chopped into packets and transmitted by a spread route over the network possibly experiencing a faulty link, which is the same as the communication via an

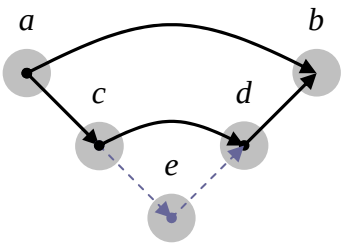
erasure channel with a random loss pattern, since the routers allocate the packets to the outgoing links randomly (according to the routing pattern). The theoretically rateless random linear fountain code [MacKay05], or the practical LT code [Luby02], or the improved Raptor code [Shokrollahi04], all reach in the file transfer application the Shannon capacity universally for any loss pattern (LT and Raptor codes with the decoding failure probability of 10^{-15}, but lower in the recent versions of Raptor). We can maintain the constant receive rate P_R by maintaining the packet transmission variable rate at $P_T = \frac{1}{1-p} \cdot P_R$, where p is the current random loss probability. At the same time the above cited fountain codes ensure the successful decoding (with a very little overhead) because they are simultaneously near-optimal for any erasure channel. Thus the routing parameter ARON for a file transfers application using at the sender a rateless universal fountain code (video chopped into files or very large blocks) and maintaining at the receiver constant reception rate (the bottleneck of the last kilometer), is computed by the following equation:

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

Let us compute the value of ARON for two routing examples over the same network shown in the figure below, where *a* is the sending node and *b* is the receiving node. Let the constant part of the tolerance of the media stream in this example be 0.1, i.e. the media stream (without additional adaptive tolerance demanded from the receiver) permanently survives the loss of up to 10%.

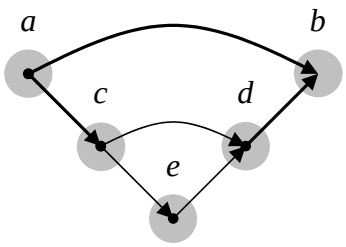


According to the first routing, shown in the figure below, the flow is evenly split over two paths $\{(a,b)\}$ and $\{(a,c),(c,d),(d,b)\}$. The ARON of this routing is equal to 3.2.



link	load	redundancy need
a *b*	0.5	0.8
a *c*	0.5	0.8
d *b*	0.5	0.8
c *d*	0.5	0.8
c *e*	0	0
e *d*	0	0

The second routing is similar to the first one except that it is slightly refined by breaking the load on the link (c,d) across itself and the two-link alternate path $\{(c,e),(e,d)\}$. The ARON of this new routing is 3, which is lower the ARON of the previous routing and therefore the new routing is FEC friendlier and is more advantageous.



link		load	redundancy need
a	*b*	0.5	0.8
a	*c*	0.5	0.8
d	*b*	0.5	0.8
c	*d*	0.25	0.2
c	*e*	0.25	0.2
e	*d*	0.25	0.2

V. Real-Time streaming and choice of an encoder

For the real-time media streaming with a limited buffering time and therefore relatively small number *M* of media packets carried in a transmission block, the required length of the transmission block cannot be derived from *M* assuming the Shannon capacity, i.e. $FEC_p \neq \frac{M}{1-p}$. Hence we must suggest a specific encoder and propose the number of media packets *M* to be carried by the transmission block. Smaller is *M* shorter is the buffering delay, important for the real-time media, but also higher will be the cost of FEC overhead. Before playing, the receiver must hold in the buffer enough packets to restore the recoverable losses. The receiving side of the media application is already equipped with a playback buffer to compensate the network jitter and to reorder packets arriving in wrong order. Thus, if necessary the playback buffer must be extended enough to consider also the packet holding resulting from the decoding needs.

Both [Huang05] and [Johansson02] (for combining with the AMR source coding) adopted simple FEC schemes: a packet level XOR and duplicates. For adding redundancy [Huang05] has proposed for every four media packets three modes with one, two and four redundant packets. In the first mode, the redundant packet is the XOR product of the four media packets. In the second mode, the redundant packets are: the XOR product of the first two media packets, and the XOR product of the last two media packets. In the third mode, four redundant packets are simply the duplicates of the four media packets. [Johansson02] proposed to extend the size of the packet with the copy of the previous packet or an XOR product of the two previous packets (or their corresponding samples encoded at lower rate).

A little more elaborated erasure resilient codes however can provide a much stronger protection. According the third FEC mode of [Huang05] a duplicate packet is sent for every media packet. If for the two original media packets *a* and *b*, i.e. out of four transmitted packets, we receive only two packets, there is a chance of about 33% that one of the packets will left unrecoverable: when you receive *a* (or *b*) with its own duplicate. The situation can be however improved, if we transmit *a* as the first redundant packet and

$a \oplus b$ (operation $\oplus$ is the binary XOR) as the second redundant packet. Under the same loss scenario, where you receive only two packets out of four transmitted, the chance that one of the packets will not be recovered is lower, but still exists and is of about 17% (that is when you receive *a* with its own duplicate). Yet another improvement is possible. We can split the packet *a* into two equal portions $a = (a_1, a_2)$. If the packet size was 20 bytes we will have thus 10 bytes in each of the halves a_1 and a_2. Similarly we break the second packet into two halves: $b = (b_1, b_2)$. As before we transmit the two original packets *a* and *b* unchanged, and additionally two redundant packets. This time the first redundant packet is $(a_1 \oplus b_1 \oplus b_2,\ a_2 \oplus b_2)$ and the second redundant packet is $(a_1 \oplus b_1,\ a_2 \oplus b_1 \oplus b_2)$. With such a FEC, we can now observe that under the same loss scenario (two packet losses out of four transmitted) we always can recover the two original media packets from any two available packets.

The author of [Nguyen03] has chosen a Reed Solomon RS(30,23) code with 23 data packets and 7 redundant packets for each FEC block. Hence, similarly to the last example above, the lost packets can be entirely recovered if there are no more than 7 losses of any type of packets.

Similarly to [Nguyen03] in our choice of an encoder we rely on any Maximum Distance Separable (MDS) erasure resilient code, Reed-Solomon is an example. We however do not fix the FEC block length, and will use different RS codes for the changing loss patterns. The small example above was an MDS code and is denoted as RS(4,2) pointing on 2 media packets in a FEC block of 4 packets. RS(4,2) is an extended Reed Solomon code, since if a packet size (or here the same – the symbol size) *s* is reduced to only two bits (not less, because in our method we must split the packets into two halves), then the classical Reed Solomon codeword length cannot be longer than $2^s - 1$ (codeword length, number of symbols or packets, all are the same here). A systematic erasure resilient MDS code adds the redundant packets along the original media packets, which are sent unchanged. It is a good choice for real-time media, because, when there are too many losses such that the FEC block cannot be decoded, a few survived original media packets, if any, could be the media fragments, still being useful e.g. via passive error concealment or the human perception. MDS is not a rateless code, like LT [Luby02] or Raptor [Shokrollahi04], but its rate range is largely sufficient for real-time streaming. RS algorithm based on 8 bits symbols is widely used, and can be scaled on packets producing FEC blocks with 255 packets, which is 10 times longer than our maximum guess of the number of media packets in a block. A possibility of a rate increase of 10 times is more than sufficient.

VI. Choice of the MDS FEC block size

By the choice of an MDS code, the reception of the sufficient number of any type of transmitted packets, precisely said exactly the same number as there were media packets in the original data, is the only condition for the successful decoding of all original media packets.

One parameter of streaming is *M*, the number of media packets in the block, another parameter we must rely on for computing the transmission block size FEC_p is the desired decoding failure probability at the receiver.

If we have 20 media packets in each transmission and 20% losses in the network, the mean number of received packets is 16. If we add 5 redundant packets and transmit for each 20 media packets a block of 25 packets the mean number of successfully received packets will be 20, which is the number needed for the recovery of the original media. However the probability of receiving 19 packets or 18 packets (which makes the decoding impossible) is quite high. Therefore for small values of *M*, which is the limiting factor of the short round-trip time of the two-way real-time media, we must send much more redundant packets in the block than is necessary to receive mean *M* packets on the receiver side. Thus the mean of received packets should be much higher than *M* and the probability of receiving less than *M* packets must be maintained very low, i.e. $FEC_p \geq \frac{M}{1-p}$.

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

$$\binom{N}{n} \cdot p^n \cdot q^{N-n}$$

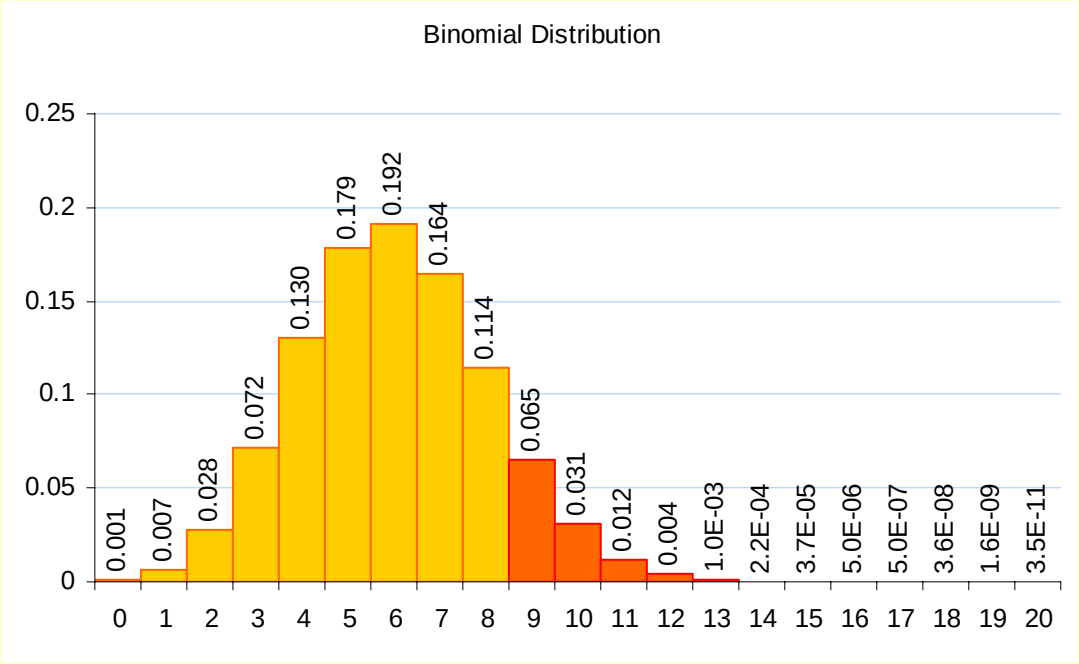
where $\binom{N}{n} = \frac{N!}{n! \cdot (N-n)!}$

and $q = p - 1$

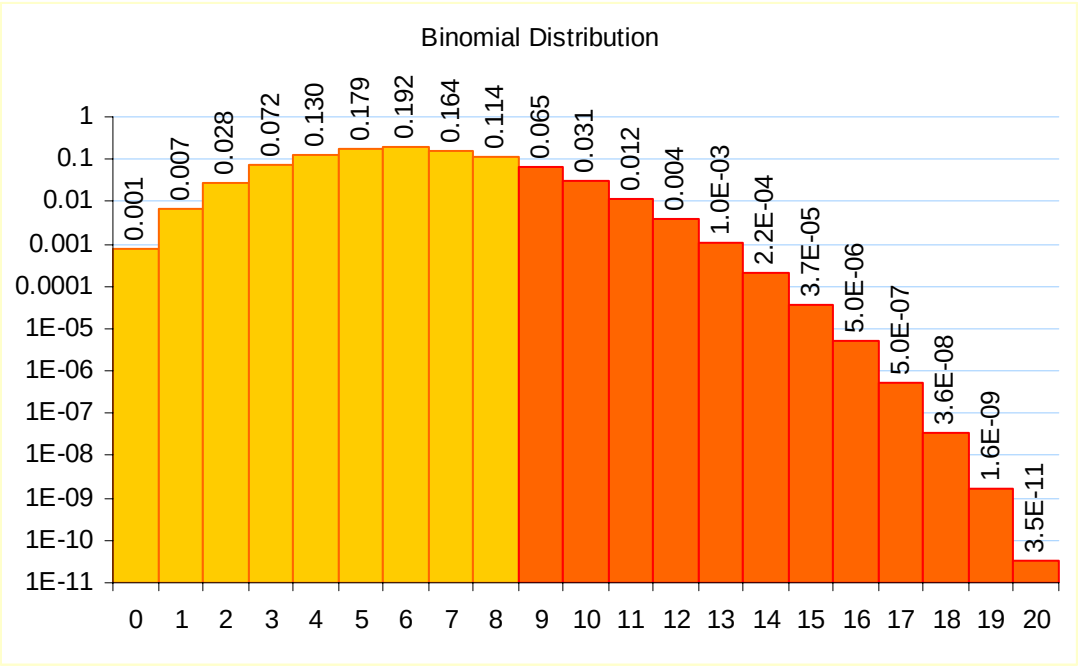
The probability of having $N - M + 1$ and more losses (i.e. less than *M* survived packets), is computed as follows:

$$\sum_{n=N-M+1}^{N} \binom{N}{n} \cdot p^n \cdot q^{N-n}$$

The below figure shows a binomial distribution for a block of 20 packets ($N = 20$) and the random loss probability of 30% ($p = 0.3$). The probability of having the mean $N \cdot p$ of losses, i.e. 6 lost packets, is the highest.



The probabilities of having 9, 10, 11 … 20 packet losses are marked on the histograms of the above chart in red. The sum of these probabilities is the probability of having more than 8 packet losses or the probability of receiving less than 12 survived packets. Below is the same chart but in the logarithmic scale.



If the number of media packets carried by a block is 12 and the number of all packets in the transmission block is 20, i.e. we use RS(20,12), then the probability of the failure of decoding at the receiver is $\sum_{n=9}^{20}\binom{20}{n}\cdot 0.3^{n}\cdot 0.7^{20-n}$.

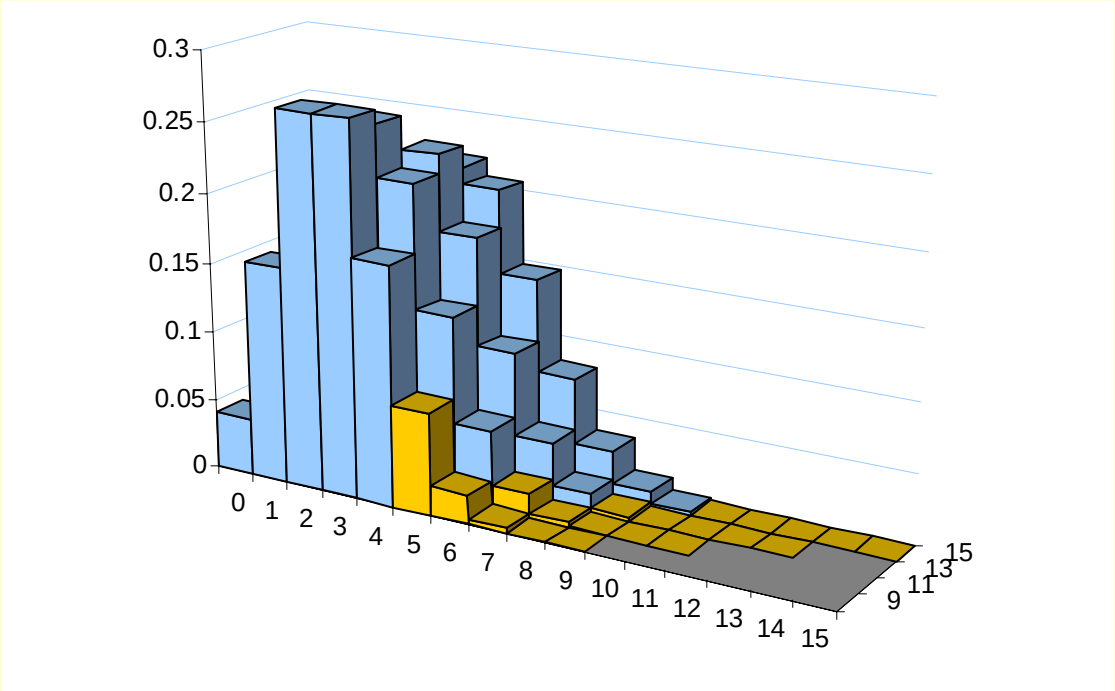
The below table contains the value of the binomial distribution of this example. Critical loss numbers are marked in red. The decoding failure probability of the RS(20,12) code sums up to about 11.3%

Number of lost packets	The probability
0	0.000797923
1	0.006839337
2	0.027845873
3	0.071603672
4	0.130420974
5	0.178863051
6	0.191638983
7	0.164261985
8	0.11439674
9	0.065369566
10	0.030817081
11	0.012006655
12	0.003859282
13	0.001017833
14	0.000218107
15	3.73898E-05
16	5.00756E-06
17	5.04964E-07
18	3.60688E-08
19	1.62717E-09
20	3.48678E-11

Media Packets	Decoding Failure probability
12 (more than 8 lost packets are critical)	0.113331463

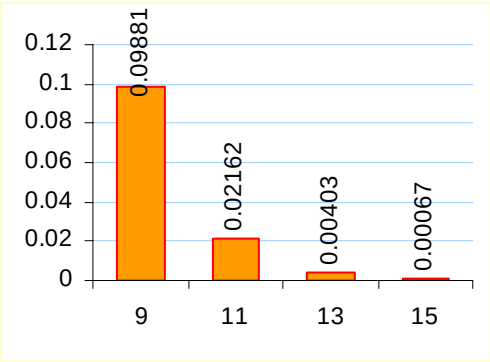
If the application sustains the decoding failure probability of 11.3%, it can then maintain for its 12 media packets the size of the transmission block at 20 packets. But if the probability of the decoding failure of 11.3% is too high, then the application needs to transmit in the carrier block more than 20 packets (for the same number of 12 media packets).

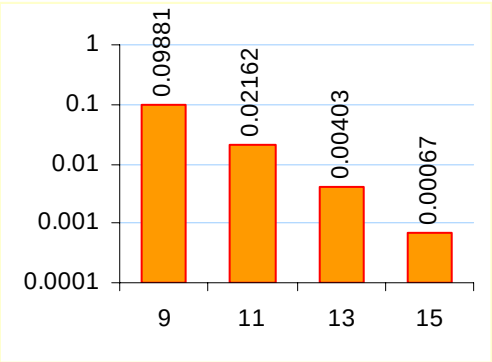
Let have another example, where *M*, the number of media packets, is equal to 5 and the desired probability of decoding failure is 1%. The probability of random loss is still 30%. The below plot shows four Binomial distributions for four carrier blocks of the lengths of correspondingly 9, 11, 13 and 15 packets.



For each of four binomial distributions, the probabilities of receiving of insufficient number of packets, i.e. only 4, 3, 2, 1 or 0 successfully received packets, we marked in yellow. The decoding is not possible in all these cases due to missing number of packets and the decoding failure probability, for each distribution, is the sum of these probabilities.

The two charts below show (in the linear and logarithmic scales) the decoding failure probability of the FEC blocks of four different sizes: 9, 11, 13 and 15 packets in length (each of these blocks carries the same number of 5 media packets).





According to the charts, if we desire to have the decoding probability below 1% we must transmit the media in blocks of 13 packets and if we wish to reduce the decoding probability below 0.1% it will be sufficient to increase the FEC block size to 15 packets.

Thus for computing the carrier block's minimally needed length for a satisfactory communication, it is sufficient to steadily increase the carrier block length until the

desired decoding error rate (DER) is met. Fast binomial distribution algorithm, especially fast algorithm for computing a value of the binomial distribution when the value at the neighboring position is already computed, and simple search optimizations quickly find the minimally required length of the FEC block at which the decoding failure probability is below or equal the permitted limit DER.

The needed length of the FEC block as a function from the random loss probability *p* has integer values. We, however, desire to have smoothly interpolated values, which do not jump from one integer value to another at slightest changes of *p*. The best fitting is obtained by a logarithmic interpolation between the following three values (under the same loss rate *p*).

- the decoding failure probability at *N*, which is below DER
- the permitted limit DER
- the decoding failure probability at $N-1$, which is higher the value of DER

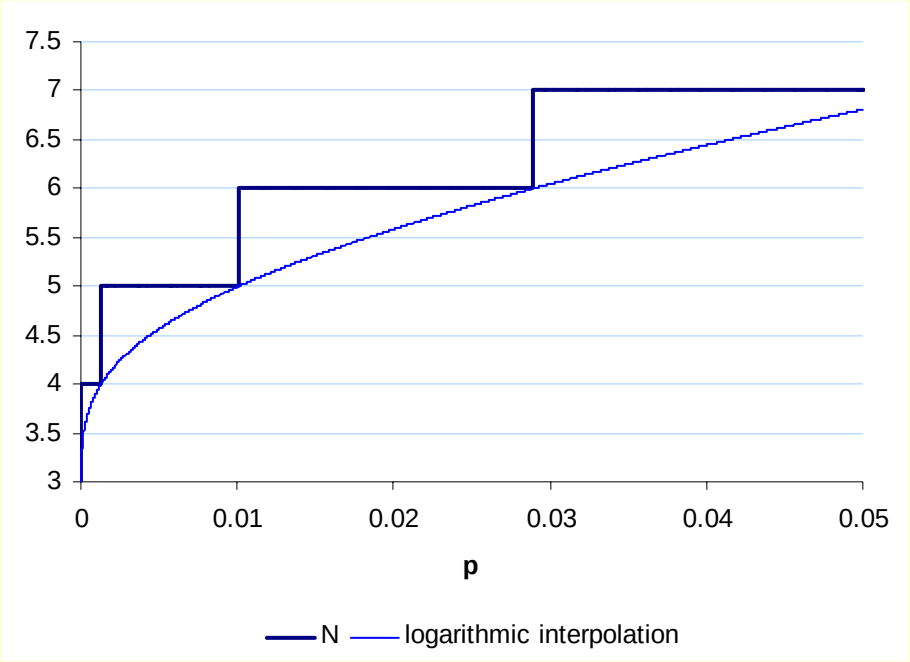
We find thus the precise FEC_p value laying between two integers $N-1$ and *N*, and corresponding "exactly" to the failure probability of DER. The below approximation of the decoding failure probability (as a function from *N*) is highly coarse, but it points to the right interpolation function (i.e. logarithmic).

$M' \cdot p'^{N-1}$ = decoding failure probability at $N-1$

$M' \cdot p'^{FEC_p} = DER$

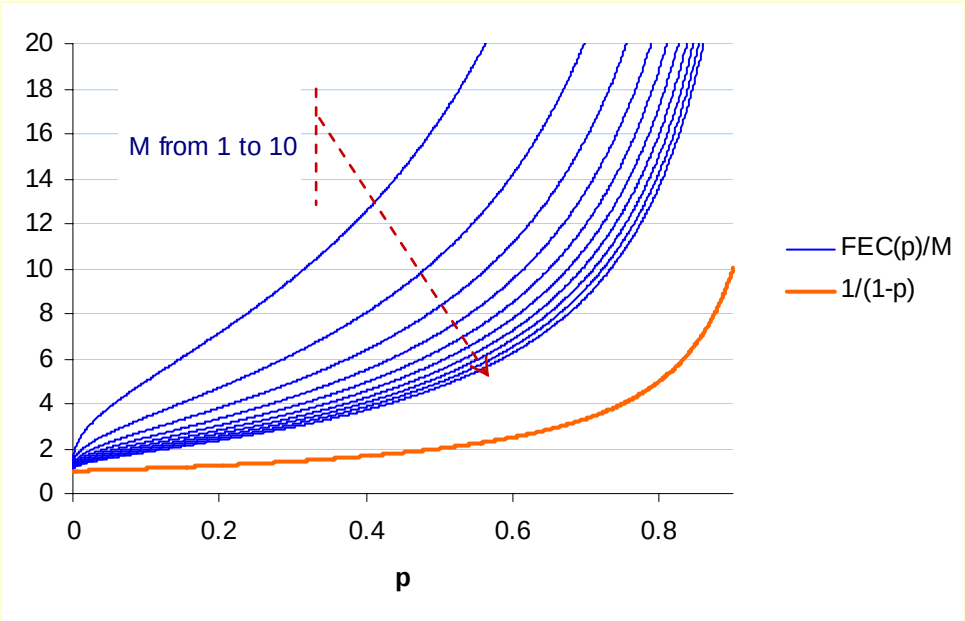
$M' \cdot p'^{N}$ = decoding failure probability at *N*

The minimal number of packets in the transmission block (an integer) needed for keeping the decoding failure probability below DER is plotted on the below chart (the thick dark blue curve). The interpolated FEC_p function is plotted on the same chart (the thin blue curve).



$DER = 10^{-5}$, $M = 3$

Comparison of several $\frac{FEC_p}{M}$ functions for media packets from 1 to 10 with the $\frac{1}{1-p}$ function, derived from the Shannon capacity, is presented in the below chart ($DER = 10^{-5}$). Higher is the number of media packets in the block (i.e. longer is the buffering time) closer the transmission will approach to the Shannon conditions.



Therefore, concluding this section: for a real-time streaming application ARON must be computed using a FEC_p function which takes into account the maximal number of media packets in the FEC block and the maximally permitted DER (as it's been presented above, see the first equation below). Assumption of the Shannon capacity (second equation below) is permitted for very long buffers, file transfers, on-line but one-way video, etc.

$$ARON = \sum_{l \in L \mid t \le r(l) < 1} \frac{FEC_{r(l)} - FEC_t}{FEC_t}$$

(1) Based on FEC_p function assuming the decoding failure probability and an MDS encoder

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

(2) Assuming Shannon capacity (file transfers)

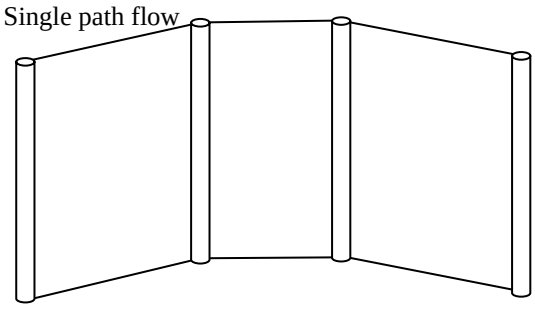
Obviously, for a given routing patter, the ARON value computed according the first equation above will be much higher the ARON value computed, according the second equation. The choice of the equation must be based on the type of the media.

We are now equipped with a technique for measuring the friendliness of a network routing toward the efforts of a streaming application in protecting its media against losses. Within a number of suggested network routings, the best is the one, whose ARON is the smallest.

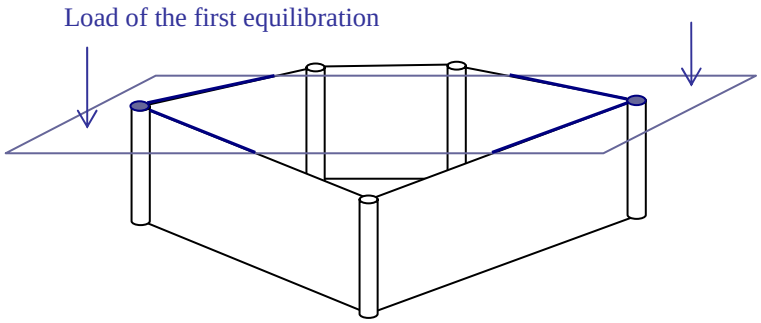
VII. Capillary Routing

Already its name suggests that this is a kind of dispersed communication flow covering numerous paths. The definition of the capillary routing and its layers is in the description of an iterative process transforming a simple flow into a capillary routing.

Capillary routing discovers the alternate paths by delegating the load of a single path route to other links. It also reaches the load balance by minimizing the upper bound value of the flow for all links. In the first layer the full mass of the flow is broken across a few parallel highways. The parallelization (spreading) factor at the first layer is an integer. If there are places where the entire traffic must pass through a single link the spreading factor will be one.

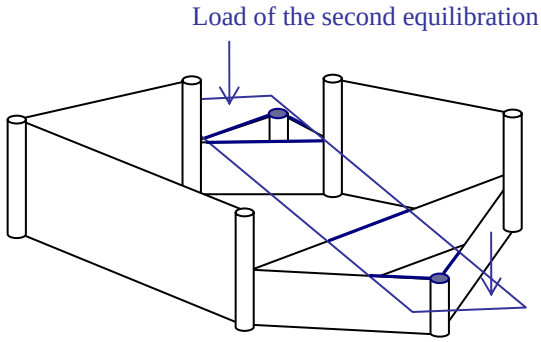


The shortest path solution



After application of the objective discovering the first layer

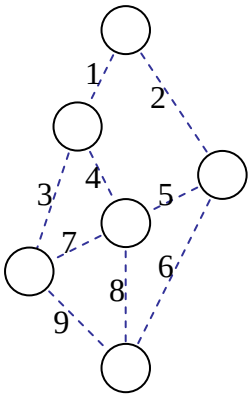
By maintaining the flow below the upper bound obtained in the first layer, further equilibration is being applied to the remaining portion of the flow. The remaining flow is additionally equilibrated by minimizing the value of the second upper bound applied on all links still maintaining a degree of freedom (links that are not the bottlenecks of the previous layer).



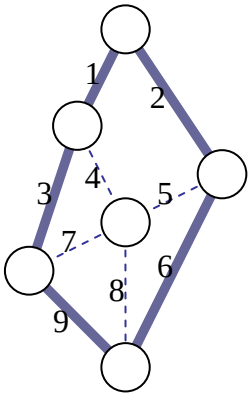
The new objective (with the constraints maintaining the previous objective) discovers the footprint of the second layer of the capillary routing

Thus we discovered the main routes of the second layer. The second layer's upper bound is added as an additional constraint (always maintaining the first constraint) and the discovering of the successive layers is continued. The capillary routing discovery is continued iteratively until the entire footprint of the flow is discovered and completely equilibrated.

Let us build the capillary routing, layer by layer on the network of the below example with 7 nodes. The top node of the diagram is the sending node and the bottom node is the receiving node. All links has descending direction.



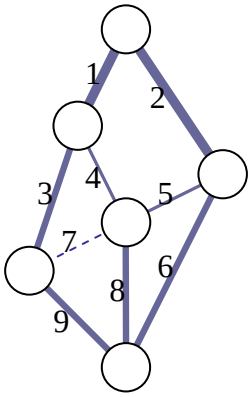
We will compute also the ARON rating for each layer of the capillary routing assuming Shannon capacity and that the stream constantly tolerates 10% of packet losses.



link	load	redundancy need
1	0.5	0.8
2	0.5	0.8
3	0.5	0.8
4	0	0
5	0	0
6	0.5	0.8
7	0	0
8	0	0
9	0.5	0.8

ARON=4

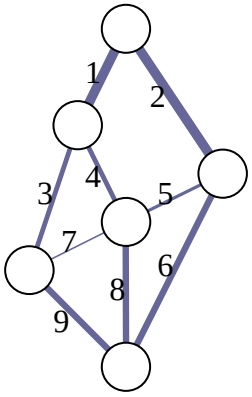
In the first layer of the capillary routing we reduce the load of bottleneck links 1 and 2 to their minimal value of $\frac{1}{2}$. Links 3, 6 and 9 are not bottleneck links and their loads can be further delegated to other links. The ARON rating of the first layer is equal to 4.



link	load	redundancy need
1	0.5	0.8
2	0.5	0.8
3	0.333333	0.35
4	0.166667	0.08
5	0.166667	0.08
6	0.333333	0.35
7	0	0
8	0.333333	0.35
9	0.333333	0.35

ARON=3.16

In the second layer we reduce the load of the current bottleneck links 6, 8, and 9 to their minimal value of $\frac{1}{3}$. The ARON rating of this layer is now 3.16. Link 3 is not a bottleneck link, and its load can be further delegated over link 4.

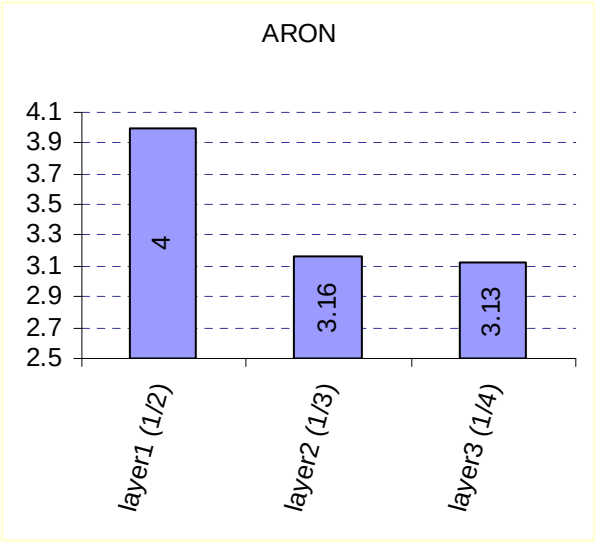


link	load	redundancy need
1	0.5	0.8
2	0.5	0.8
3	0.25	0.2
4	0.25	0.2
5	0.166667	0.08
6	0.333333	0.35
7	0.083333	0
8	0.333333	0.35
9	0.333333	0.35
		ARON=3.13

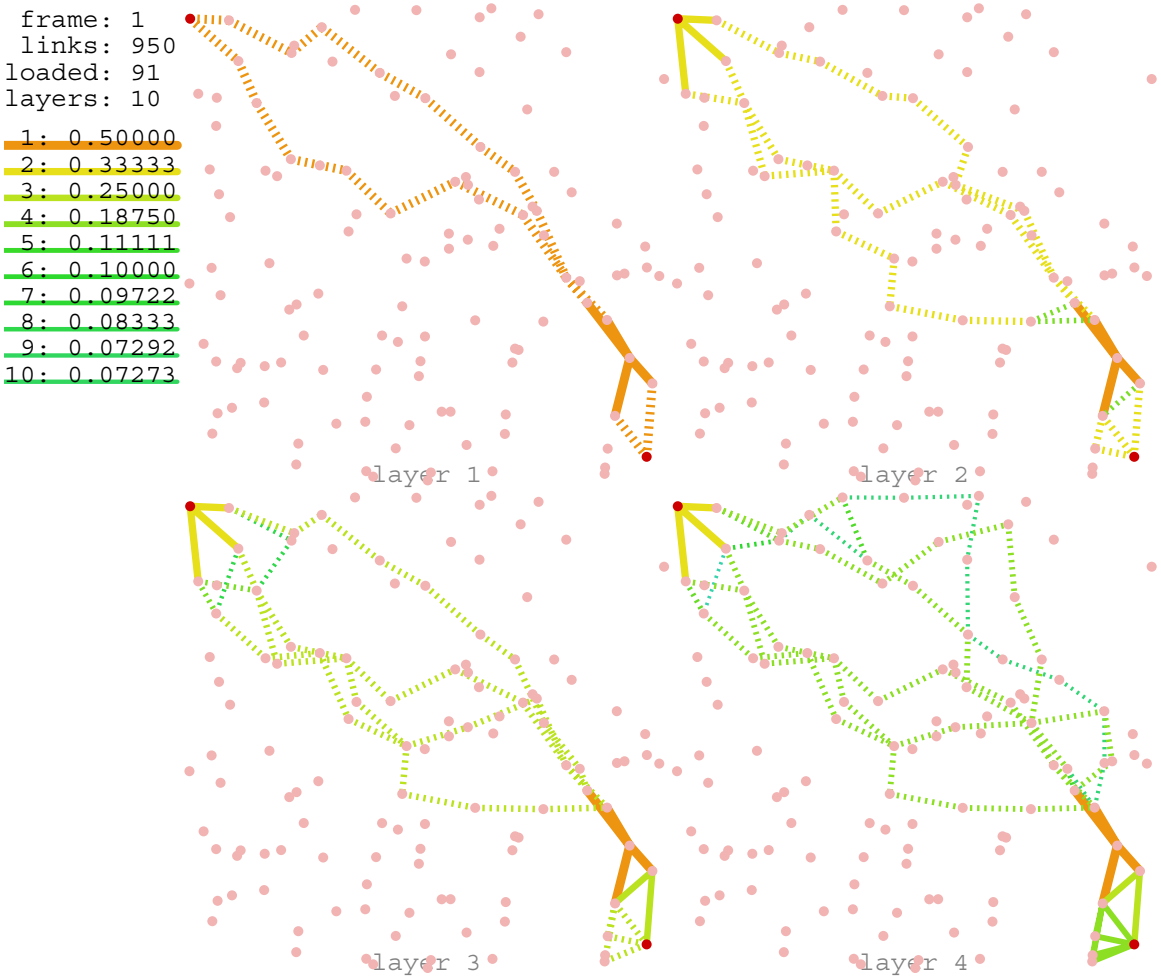
In the third layer we equilibrate the bottlenecks of the current layer, links 3 and 4. Their load is reduced to $\frac{1}{4}$.

In the last solution the load of link 5 is $\frac{1}{6}$ and the load of link 7 is $\frac{1}{12}$. There is no freedom left for the flow after the solution of the third layer, so if we follow the layering, link 5 will be the only bottleneck link of the forth layer with its load of $\frac{1}{6}$ and link 7 will be the only bottleneck of the fifth last layer with its load of $\frac{1}{12}$.

Progress of the capillary routing through its first three layers is shown in the plot below.

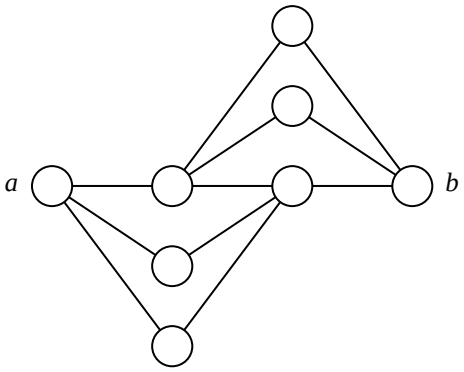


In the figure below we present an example of the capillary routing on a network with 120 nodes. We present first four layers of the capillary routing, i.e. four routing suggestions on the same network. In the first layer the flow is following two paths, thus the load for the first layer is 0.5 (see the information bar on the left from the plots). Solid lines on the path are marking the links, which are the bottlenecks of the layer, paths in dashed lines, are the links with a degree of freedom (whose load can be further delegated). The flow follows a spreader path in the second layer, whose links carry no more than $\frac{1}{3}$ of the traffic, except the bottleneck links of the first layer. The 3rd layer is yet more spreader and its links, except the few bottleneck links of the first and second layers, are carrying not more than $\frac{1}{4}$ of the traffic (follow the vertical information bar on the left side of the figure). Although only four layers are visualized, the capillary routing is computed for first 10 layers, and the loads of the layers are displayed in the vertical information bar. The entire network contains 950 links, which are not shown if no flow is routed through them. 91 is the total number of all bottleneck links in all 10 layers.

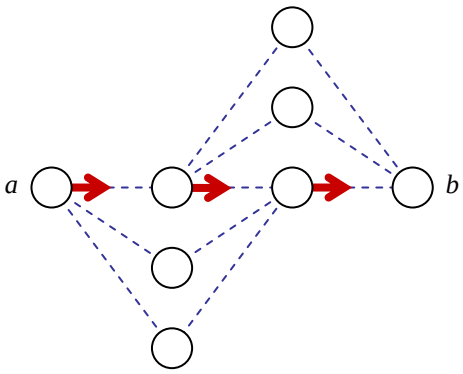


In [Ma03] it's been studied concept of hop-by-hop packet recovery with packet level FEC. In this study, the author simplified the network topology into two categories: (i) parallel and (ii) serial connection of links, assuming that a link can be abstracted into a sub-network without connectivity to other links (sub-networks). However contrary to the author's belief the generality of this routing model suffers noticeably from such simplifications. Topologies not fitting into the model of the author are especially frequent in a uniform network environment, such as MANET, which is precisely a subject of a study of the author (the above example of a network with 120 nodes is a snapshot of MANET as well).

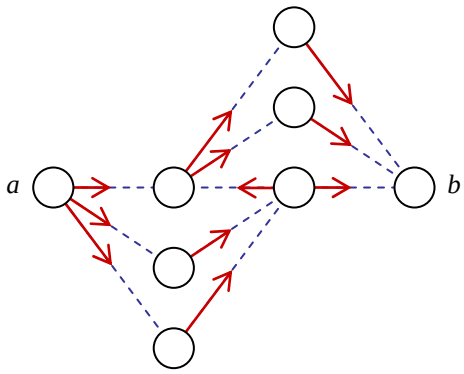
In the below diagram where *a* is the sending node and *b* is the receiving node the underlying network is a small example of the topology which is not fitting in the model of [Ma03]. Network is a symmetric directed graph, each line representing a pair of opposed arcs. The network cannot be represented simply via a sequence of parallel and serial links.



It becomes obvious, when we define different objectives on the same network. When we seek for the shortest path flow, we obtain a single path routing carrying the entire flow (the figure below).



Shortest path solution



Max-flow solution

In the max-flow objective over the same network with the same peers, the flow however may even follow on certain links, a direction opposite to that of the shortest path solution (the link between the two middle nodes of the shortest path).

VIII. Building Capillary Routing

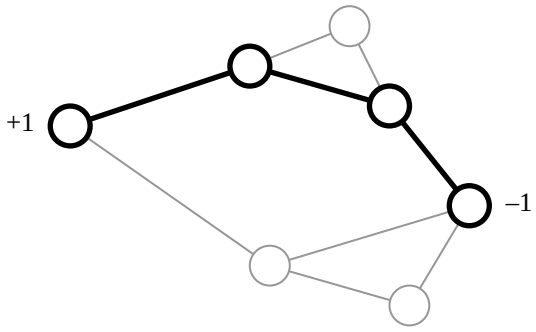
One approach for building the capillary routing is derived straight from the iterative definition of the capillary routing. With an LP model first minimize the maximal value of the load. Find the bottleneck links of the first layer. Minimize the maximal load of remaining links and identify the bottleneck links of the second layer. Remove the bottleneck links of the second layer and continue the iteration on the remaining links…

In this approach we reach quickly problems with very tiny loads, which make the LP method very sensible to precision errors. It is better to keep the values of parameters and variables always in the same order of grandeur.

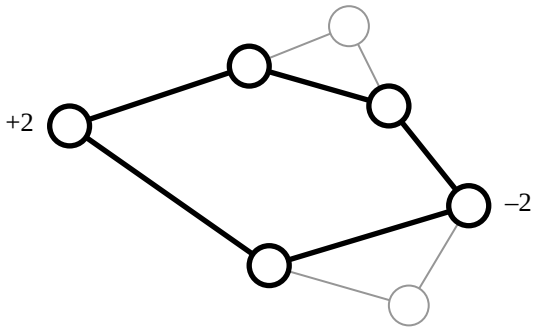
In order to keep the values of parameters and variables on the same scale we are applying a different LP method. The routing of a given layer of capillary routing is discovered by increase of the flow (solving the max flow problem) instead of decreasing the maximal value of loads of links. The resulting flows of these two methods are identical except that the proportions are different by the flow increase factor of the max flow solution.

The LP problem of the successive layer is obtained by removal of bottlenecks from the network, producing thus new sources and sinks in the network.

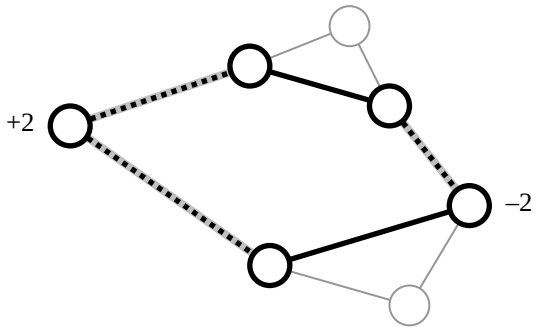
Below is an example of the capillary routing discovery process from the first layer through the max flow solution of the second layer:



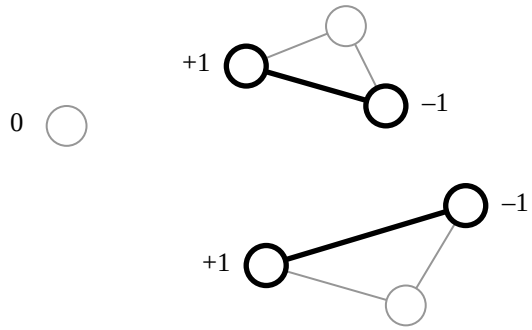
Initial problem with one source and one sink node



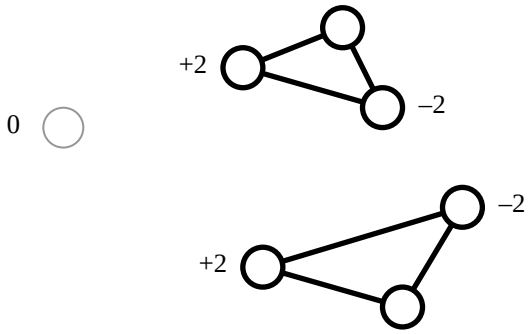
Maximally increase the flow and adjust the flow out coefficients at nodes



Identify the bottleneck links (dashed)



Remove the bottleneck links from the network adjusting correspondingly the flow out coefficients at adjacent nodes



Again solve the max flow problem (respecting a synchronous grow of flows at all sources and sinks)

Thus if the flow problem of a synchronous multiple-multicast at a layer l is defined 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 values f_i^l for all $i \in N^l$

And in its max-flow solution:

- the synchronous flow increase factor for all nodes is F^l and
- the set of bottlenecks is B^l (where $B^l \subset L^l$)

Then the new problem of the successive layer: N^{l+1}, L^{l+1} and f^{l+1} is defined as follows:

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

After certain number of applications of the max-flow 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 will be enclosed in bottlenecks of one of the layers.

Only the initial problem (at the layer 1) is definitively a unicast problem. All successive layers, in general are synchronous multiple-multicast problems (a uniform flow from set of sources to set of sinks, where all rates of transmissions by sources and all rates of receptions by sinks are proportional, via each node's proportionality coefficient, to a single variable), and thus do not belong to the simple class of "network linear programs" [Fourer03].

The absolute flow $r_{i,j}$ traversing the link $(i, j) \in B^l$ in the capillary routing must be computed according the following equation:

$$r_{i,j} = \frac{1}{\prod_{i=1}^{l} F^i}$$

where l is the layer for which $(i, j) \in B^l$

Description of the bottleneck identification was deliberately left for the end of the section, since behind this single phase are hidden solutions to several LP problems. 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 traffic over all links being suspected as bottlenecks. Only links maintaining their load high will be passed to the next iteration. Links undergoing to load reduction under the LP objective are removed from the suspect list and the iteration stops if there are no more links reducing their load.

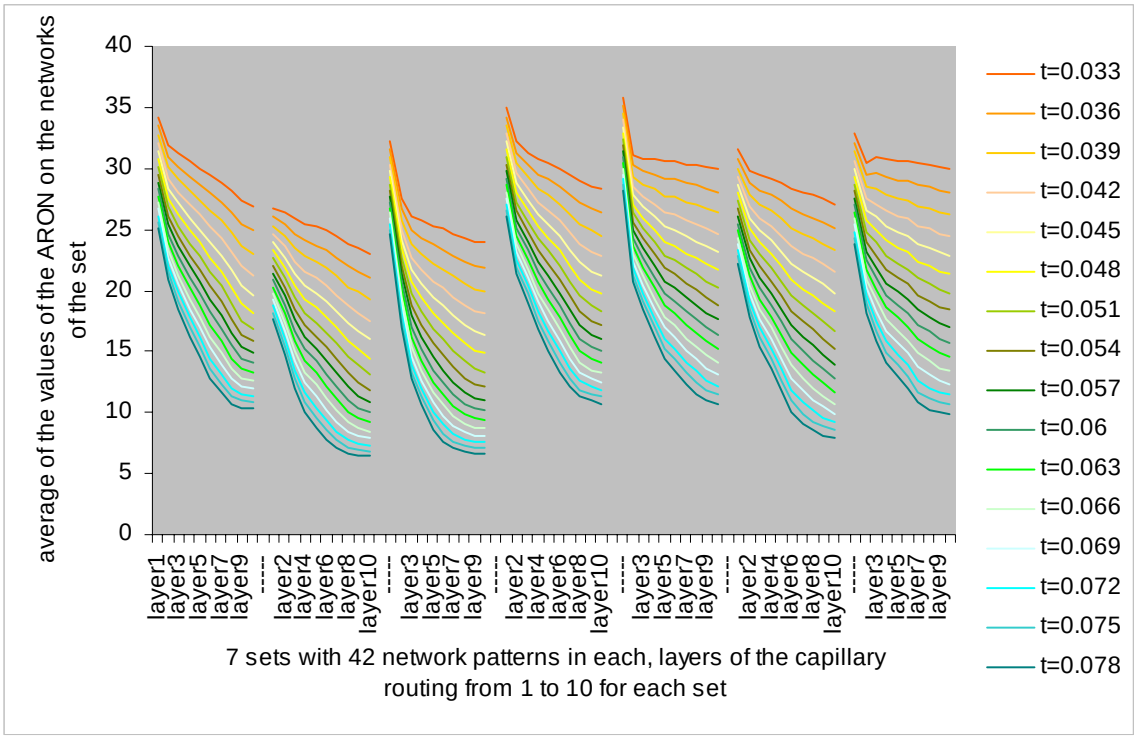
IX. ARON of Capillary Routing

To evaluate the friendliness of the capillary routing toward tolerant media streaming we must measure the ARON values of routings suggested by each capillary layer. There is no need to compare the capillary routing with any single path routing suggestion, which is too bad, since no FEC may save the media stream if the link failure time (or the failure detection and recovery time) is longer than the media buffering time. And so, as a reference for the comparison, we use the first layer of the capillary routing (with removed critical links if any). The first reference layer is a routing with a load balance of the main mass of the flow. The routing of the first layer has the footprint of the max-flow solution. The first layer of the capillary routing is similar to the path diversity schemes discussed in [Nguyen03], [Ma03], [Qu04] and [Tawan04].

To evaluate the overall performance of the capillary routing technique, rather than to measure the performance of a particular routing example, we need many network samples. Thus we can take the average ARON rating for all network samples in the test-

bed set. First we suggest the first layer routing individually to each network sample of the test-bed set and we obtain thus the average ARON rating for all routing suggestions of the first layer. Then we compute the second layer routing individually for each network sample in the set and we obtain therefore the set's average ARON rating for the routing suggestions of the second layer. We measure similarly until the average ARON for the routings of the 10th capillary layer. We obtain thus an overall performance figure toward the layers of the capillary routing for the entire set of network samples.

In the figure below we have seven test-beds, each one with 42 network samples. For each test-bed we have a curve of the average ARON, which decreases as the capillary routing layer increases from 1 thru 10. Instead of one curve there are several curves for ARON, each following the tendency of decrease with the increase of the layer. These curves represent 16 different media streams with various values *t* of permanently maintained constant tolerance to losses.

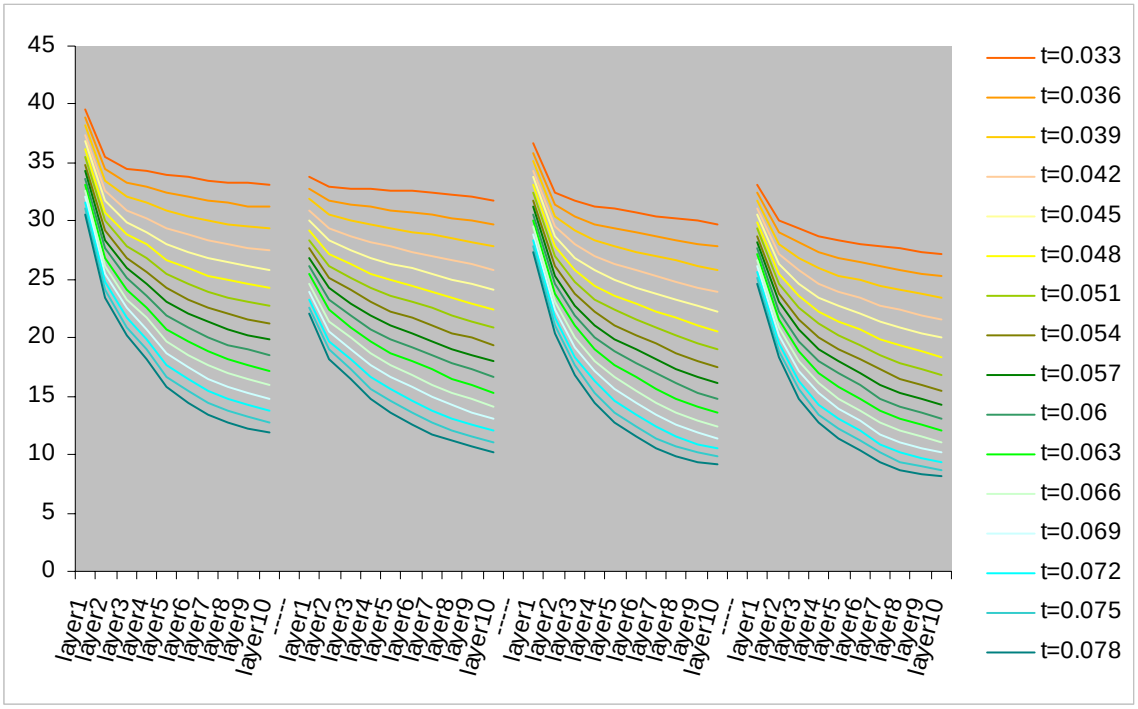


For example the permanently maintained constant tolerance of a VOIP stream, using the medium complexity (g729r8) or high complexity codecs (g723r53 and g723r63), is from 8% to 11% of packet losses (with a payloads from 20 up to 60 bytes). Although the video stream is more sensible to losses, it also is equipped with an internal tolerance resulting from the passive error concealment when the motion level is not very high and from an internal weak FEC code. The permanent tolerance of any application (even with zero initial tolerance like in a file transfer) can be obtained by constantly maintaining at the source a proper amount of the redundant data. Logically, the ARON curve of the media stream is shifted down by every unit of the added permanent tolerance. At the same time

it is interesting to observe that the presence of a permanent tolerance in the media stream also stresses the efficiency gain achieved by the deeper layers of the capillary routing.

The network samples for the test-beds are obtained from a random walk Mobile Ad-hoc Network (MANET) on a rectangular space (Several other authors studied reliable multi-path routing in MANET: [Tawan04] or [Ma03] and [Ma04] using a concept of hop-by-hop packet recovery assuming regenerating nodes). Initially our nodes are randomly distributed on the rectangular area, and then at every timeframe they move according to a random walk algorithm. If two nodes are close enough (are within the coverage range) then there is a link between them. In the above example, there are 115 nodes and 300 timeframes, and therefore 300 various network samples. These 300 patterns are broken into 7 sets represented on the above chart. The number *M* of media packets per each transmission block is 20 and the desired decoding failure rate DER is 10^{-5}. In AMR codec or in g729r8 codec with a 20 byte payload the sample duration is 20 ms, thus 20 packets in the transmission block requires a playback buffer of 400 ms. In real-time voice communication the human perception may tolerate 500 up to 600 ms delay [Padhye00].

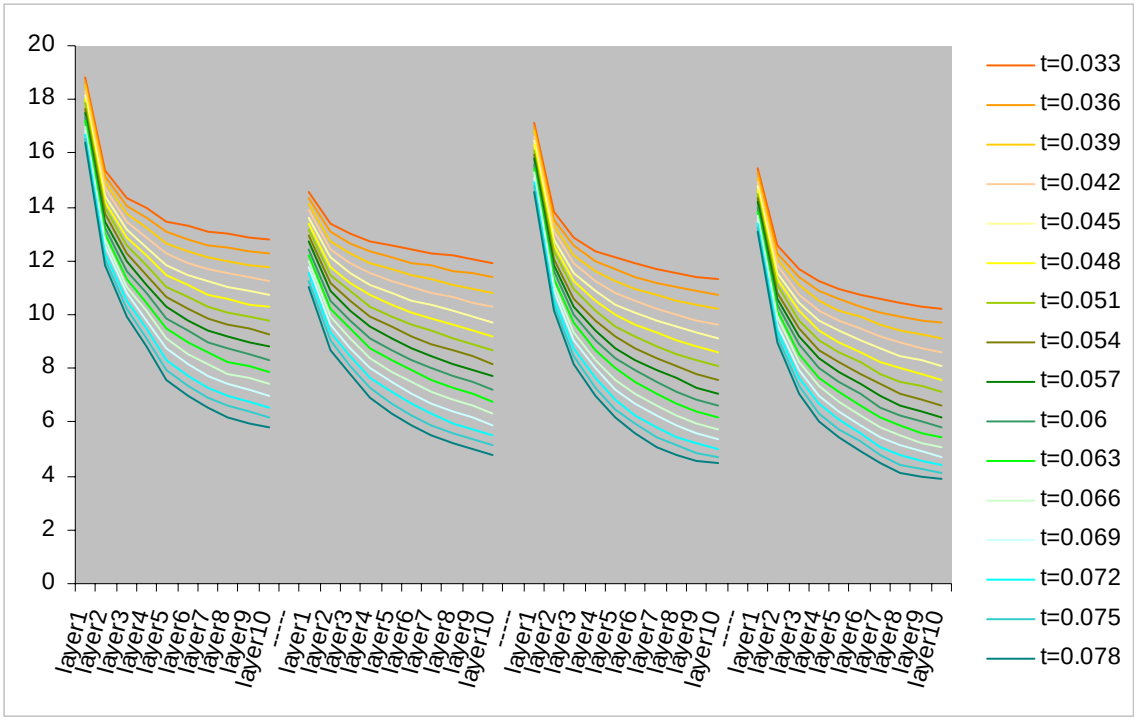
In the example below the network consists of 120 nodes and there are 150 timeframes of the random-walk ad-hoc network. The 150 different network samples are broken into four sets.



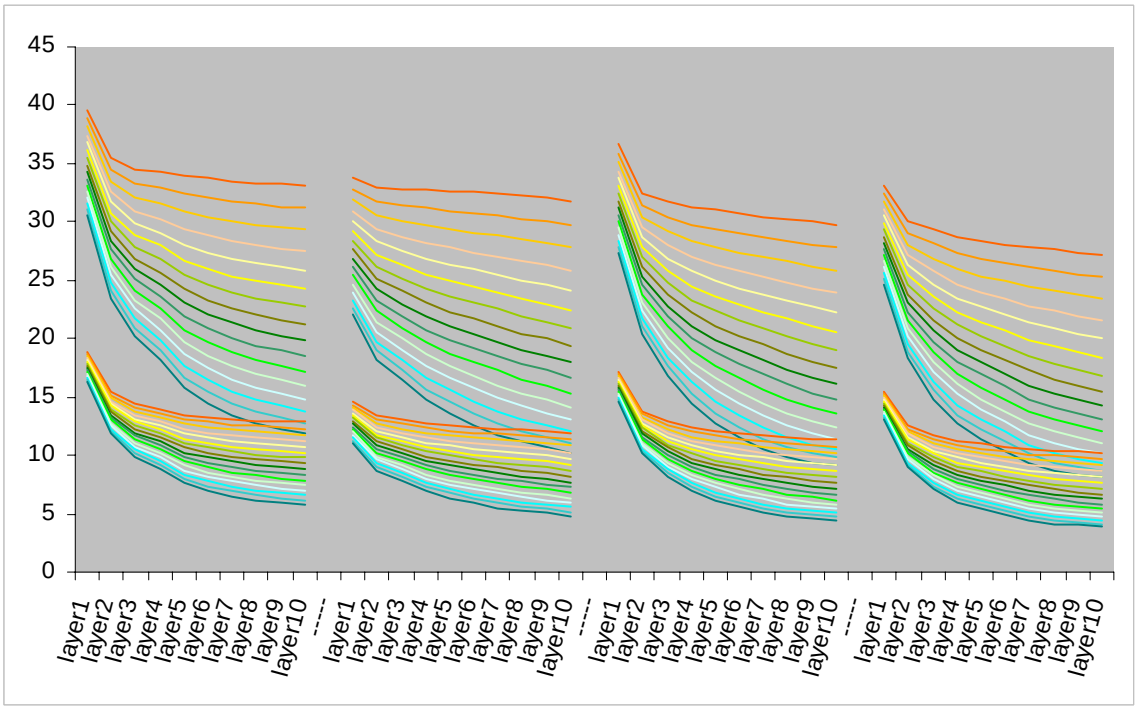
The media streaming parameters are identical to those of the previous example ($M = 20$, $DER = 10^{-5}$).

For the same set of the network samples we may wish to see how performs ARON assuming Shannon capacity. As expected ARON computed with the Shannon capacity

(see the chart below) assumption has much lower values compared with “true” ARON assuming the MDS encoder and the binomial probabilities of the failure.



Below are the both curves on one chart, “true” ARON curves above and the curves derived from the Shannon capacity below.



It remains us to show yet another chart, which does not precisely represent the ARON values but other similar values which characterizes rather the encoder effort during the failures than during the whole communication time. Recall that the formula below gives us the number of the redundant packets that the sender will be obliged to additionally send, due to the network failures, during the communication period *T*:

$$d \cdot \frac{T}{s} \cdot B \cdot FEC_t \cdot ARON$$

Where:

- *s* is the failure periodicity of a single link (the same for all links), i.e. the link fails approximately every *s* seconds
- *d* is the duration of a single failure
- *B* is the block transmission rate (which is supposed to be constant, while the packet transmission rate may rise due to adaptive increase of the redundant packets in the block)

ARON, represented by the below equation, is the proportionality coefficient for the amount of redundancy transmitted during the communication time.

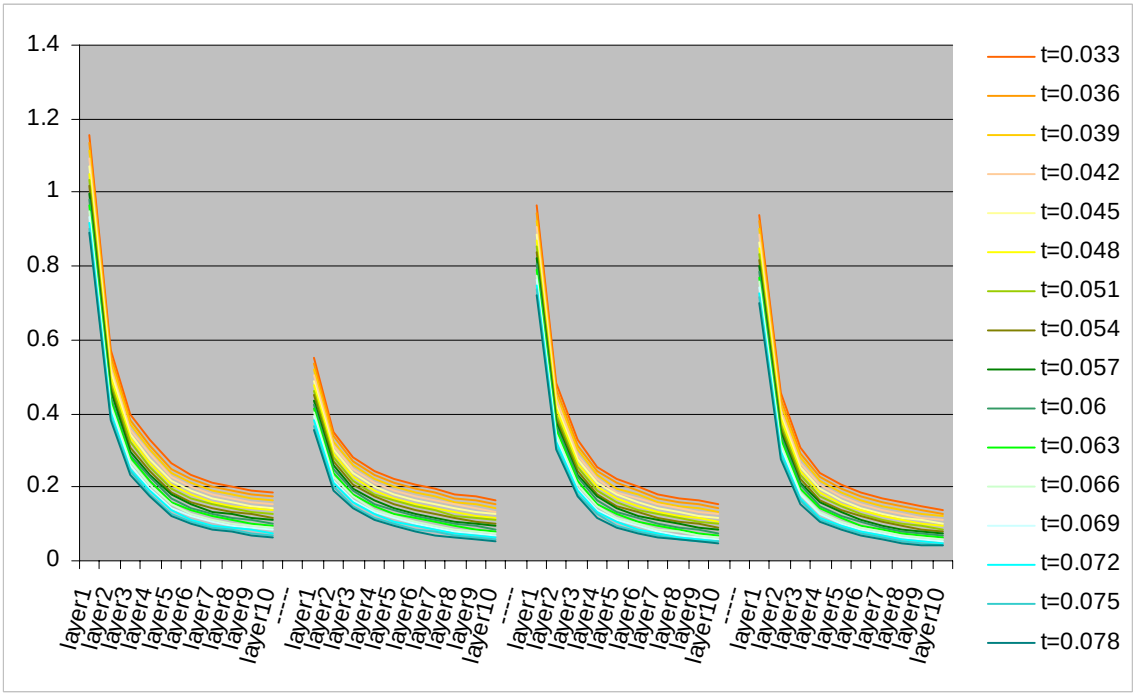
$$ARON = \sum_{l \in L \mid t \le r(l) < 1} \frac{FEC_{r(l)} - FEC_t}{FEC_t}$$

By spreading the communication footprint (when we increase the layer of the capillary routing) we increase the frequency of link failures, however the overall value of ARON goes down, since the effort necessary to recover an increased number of failures of lightly loaded links is lower the effort needed for the recovery of a few failures but of highly loaded links. In [Nguyen03] the author used RS(30,23) code and studied the routing of media through an OSPF route (with 10 ms failure time per 1 second of good time) altered with a longer route of five times lower quality (50 ms failure time). It was shown that the double-path routing, even if involving a bad route, is better the single shortest path routing. According to the author, this suggests that if there exists a redundant path which is not “nearly” as good as the default path between sender and receiver, then it is still advantageous to deploy the path diversity scheme.

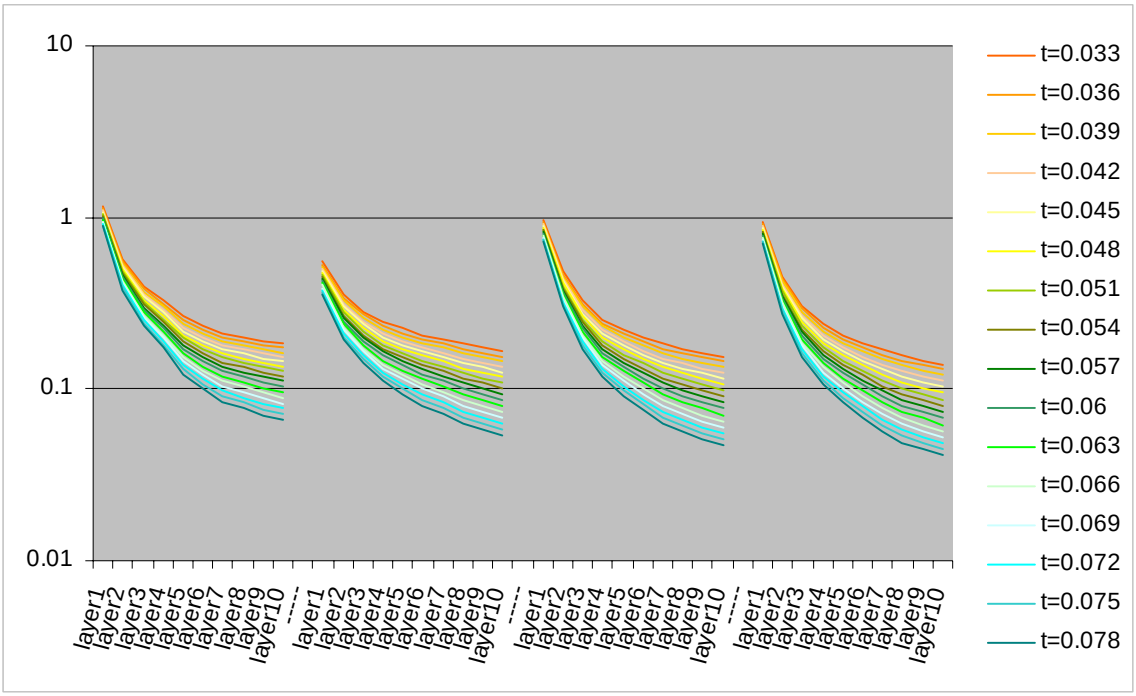
However one may not be interested in the total amount of the sender’s encoding effort during the whole communication time, but in the average effort required from the encoder only during the failure time. The average effort expected from the sender whenever the receiver reports a failure (by observing missing packets) may have higher significance for a particular application and can be computed as follows:

$$\frac{\sum_{l \in L \mid t \le r(l) < 1} \frac{FEC_{r(l)} - FEC_t}{FEC_t}}{\sum_{l \in L \mid r(l) < 1} 1}$$

The two charts below show the progress of this value, in the linear and the logarithmic scales, as the layers of the capillary routing rise. We can observe that the average recovery cost per link failure decreases more than 10 times from the first layer of the capillary routing through its 10th layer.



Average cost of single link failure recovery in the linear scale for the first 10 layers of the capillary routing



Average cost of single link failure recovery in the logarithmic scale for the first 10 layers of the capillary routing

X. Implementation suggestions

The results of our research are expected to have some impact on the development of explicit multi-path routing strategies. We hope that our investigation will provide some guidelines for future design of diversity-based multimedia transmission systems.

As for immediate use, except for cases where the application and the network is under the control of the same company, it will be too naïve to think that ISP routers can collaborate in any way with the end user streaming application. Also the existing streaming applications themselves are not assuming any engine for interfacing with the network routing (except probably the QoS flags).

Since changes of the routing of the physical network are restricted, we suggest obtaining capillary routing on overlay network using relay nodes or media routers [Nguyen03]. As a practically feasible application we can envisage relatively cheap UDP media proxy routers redirecting the UDP traffic on the application level. An ITSP, may collocate or host its media gateways in various network places, especially beneficiary is the hosting in premises of those ISPs who connects a noticeable numbers of the ITSP's clientele. A hardware or software layer between the user streaming application and the network may be in charge of spreading the flow from the user agents (UA) to the destination, via the media gateways. The media gateways can be combined with the functionality of SIP (Session Initiation Protocol) servers or proxies. The UA layer can be implemented in the firmware of a SIP phone, in the NAT router of the user or in the closest SIP proxy.

Spreading can be obtained at a lower network layer, still controlled by end user and ITSP. ITSP can use VPN tunnels to spread the communication footprint. The flow at the source can be split across various VPN virtual interfaces. Since the VPN servers, forming an overlay network, are scattered, the flow of encapsulated packets will necessarily follow a wide footprint until is reached the VPN servers. Further, once the packets are released by the VPN servers to the open internet, the flow will be routed to the destination, still following a spread footprint.

Finally the ISP, if wishes to be real-time media tolerance friendly, can structure the routing of its network to be statically spread. Most routers, including all recent IOS releases of Cisco, provide packet level spread routing, EIGRP or balanced static routing. To combat packet loss along unreliable paths it is advisable also to use EIGRP in the packet load balance mode instead of the session load balance mode.

XI. References

[Altman01] Eitan Altman, Chadi Barakat, Victor M. Ramos, "Queueing analysis of simple FEC schemes for IP telephony", Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 2001, Volume 2, April 22 – 26, pages 796 – 804

[Byers99] John W. Byers, Michael Luby, Michale Mitzenmacher, "Accessing multiple mirror sites in parallel: using Tornado codes to speed up downloads", Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 1999, Volume 1, March 21 – 25, pages 275 – 283

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

[hollywood03] Mark Fritz, "Digital Dailies Flow Freely from Fountain", April 1, 2003, http://www.emedialive.com/Articles/ReadArticle.aspx?CategoryID=45&ArticleID=5077

[honda04] Loring Wirbel, "Deal pushes algorithms into digital radio", April 13, 2004, http://www.commsdesign.com/showArticle.jhtml?articleID=18901216

[Huang05] Yicheng Huang, Jari Korhonen, Ye Wang, "Optimization of Source and Channel Coding for Voice Over IP", Multimedia and Expo 2005, ICME'05, July 06, pages 173 – 176

[Johansson02] Ingemar Johansson, Tomas Frankkila, Per Synnergren, "Bandwidth efficient AMR operation for VoIP", Speech Coding 2002, IEEE Workshop, October 6 – 9, pages 150 – 152

[Kang05] Seong-ryong Kang, Dmitri Loguinov, "Impact of FEC overhead on scalable video streaming", ACM Network and Operating System Support for Digital Audio and Video 2005, NOSSDAV'05, June 12 – 14, pages 123 – 128

[Krunz04] Marwan M. Krunz, Mohamed Hassan, “Adaptive rate control scheme for video streaming over wireless channels”, Data Compression Conference 2004, DCC 2004, pages 242 – 251

[Luby02] Michael Luby, “LT codes”, The 43rd Annual IEEE Symposium on Foundations of Computer Science 2002, FOCS’02, November 16 – 19, pages 271 – 280

[Ma03] Rui Ma, Jacek Ilow, “Reliable multipath routing with fixed delays in MANET using regenerating nodes”, 28th Annual IEEE International Conference on Local Computer Networks 2003, LCN’03, October 20 – 24, pages 719 – 725

[Ma04] Rui Ma, Jacek Ilow, “Regenerating nodes for real-time transmissions in multi-hop wireless networks”, 29th Annual IEEE International Conference on Local Computer Networks 2004, November 16 – 18, pages 378 – 384

[MacKay05] David J. C. MacKay, “Fountain codes”, Communications, IEE Proceedings Volume 152 Issue 6, December 2005, pages 1062 – 1068

[MBMS05a] Technical Specification Group Services and System Aspects, “Report of FEC selection for MBMS”, for information, March 14 – 17, 2005, http://www.3gpp.org/ftp/tsg_sa/TSG_SA/TSGS_27/Docs/PDF/SP-050088.pdf

[MBMS05b] Technical Specification Group Services and System Aspects, “Efficient FEC code for reliable MBMS user services”, for discussion and approval, March 14 – 17, 2005, http://www.3gpp.org/ftp/tsg_sa/TSG_SA/TSGS_27/Docs/PDF/SP-050164.pdf

[MBMS05c] Technical Specification Group Services and System Aspects, “Report of MBMS FEC Status”, for information and action, June 6 – 9, 2005, http://www.3gpp.org/ftp/tsg_sa/TSG_SA/TSGS_28/Docs/pdf/SP-050246.pdf

[Nguyen02] Thinh Nguyen, Avideh Zakhor, “Protocols for distributed video streaming”, Image Processing 2002, Volume 3, June 24 – 28, pages 185 – 188

[Nguyen03] Thinh Nguyen, P. Mehra, Avideh Zakhor, “Path diversity and bandwidth allocation for multimedia streaming”, Multimedia and Expo 2003, ICME’03 Volume 1, July 6 – 9, pages 663 – 672

[Padhye00] Chinmay Padhye, Kenneth J. Christensen, Wilfrido Moreno, “A new adaptive FEC loss control algorithm for voice over IP applications”, Performance Computing and Communications Conference 2000, IPCCC’00, February 20 – 22, pages 307 – 313

[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”, Global Telecommunications Conference 2004, IEEE GLOBECOM’04 Volume 2, November 29 – December 3, pages 977 – 981

[Shokrollahi04] Amin Shokrollahi, “Raptor codes”, International Symposium on Information Theory 2004, ISIT’04, June 27 – July 2, page 36

[Tawan04] Tawan Thongpook, “Load balancing of adaptive zone routing in ad hoc networks”, TENCON 2004, Volume B, November 21 – 24, pages 672 – 675

[Weatherspoon02] Hakim Weatherspoon, John D. Kubiatowicz, “Erasure Coding vs. Replication: A Quantitative Comparison”, First International Workshop on Peer-to-Peer Systems, IPTPS’02, March 7 – 8, pages 328 – 337

[Xu00] Youshi Xu, Tingting Zhang, “An adaptive redundancy technique for wireless indoor multicasting”, Fifth IEEE Symposium on Computers and Communications 2000, ISCC 2000, July 3 – 6, pages 607 – 614

XII. Glossary

3G	3rd Generation mobile communication
3GPP	3rd Generation Partnership Project
ADSL	Asynchronous Digital Subscriber Line
AMR	Adaptive Multi-Rate voice codec 4.75 - 12.2 kbps
ARON	Adaptive Redundancy Overall Need
ARQ	Automatic Repeat reQuest
BER	Bit Error Rate
CPU	Central Processing Unit
DER	Decoding Error Rate
DoS	Deny of Service
EIGRP	Enhanced Interior Gateway Routing Protocol
FEC	Forward Error Correction
FIFO	First In, First Out
g723r53	High complexity voice codec G.723.1 5300 bps
g723r63	High complexity voice codec G.723.1 6300 bps
g729r8	Low complexity voice codec G.729 8000 bps
gsmfr	High complexity voice codec GSMFR 13200 bps
HTTP	HyperText Transfer Protocol
IOS	Internet Operating System
IP	Internet Protocol
ISP	Internet Service Provider
ITSP	Internet Telephony Service Provider
LP	Linear Program
LT	Luby Transform Code
MANET	Mobile Ad-hoc Network
MBMS	Multimedia Broadcast/Multicast Service

MDS	Maximum Distance Separable
MPEG	Moving Picture Experts Group
NAT	Network Address Translation
QoS	Quality of Service
RS	Reed-Solomon
RTP	Real-time Transport Protocol
RTT	Round Trip Time
SIP	Service Initiating Protocol
TDM	Time Division Multiplexing
UA	User Agent
UDP	User Datagram Protocol
VOIP	Voice Over IP
VPN	Virtual Private Network
XOR	eXclusive OR

XIII. Relative links

	Mirrors	
This document	US	Swiss
Web version of this article	US	Swiss
Releases of capillary routing builder and ARON measurer	US	Swiss
ARON measurer code	US	Swiss
Capillary routing code	US	Swiss
Examples of capillary routing	US	Swiss
More examples	US	Swiss
Computation of the MDS FEC block size needed for the recovery of given random losses at a desired decoding failure rate	US	Swiss
Bottleneck hunting example in synchronous multicast	US	Swiss