SCALABLE VIDEO DISSEMINATION WITH PRIORITIZED NETWORK CODING

Eymen Kurdoglu, Nikolaos Thomos and Pascal Frossard

Signal Processing Laboratory (LTS4)
Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland
{eymen.kurdoglu,nikolaos.thomos,pascal.frossard}@epfl.ch

ABSTRACT

In this paper, we present a pull-based dissemination protocol for efficient distribution of scalable video content in overlay peer-to-peer networks with mesh structures. The proposed protocol employs prioritized network coding, where the network coded packets belong to classes that represent packets of different priorities. For a receiver, the pull procedure begins with the reception of *buffer vector* messages from the senders, which bring information about the numbers and classes of available packets. The receiver node decides on the rate allocation of the different classes to be requested from each of the senders. The rate allocation is cast as a video quality maximization problem and solved using a hill-climbing algorithm. The simulation results show that the proposed mechanism, which is able to fully adapt to network dynamics, accounts for the unequal packet importances and utilizes the network resources efficiently.

Index Terms— Network coding, mesh, video streaming, unequal error protection, peer-to-peer

1. INTRODUCTION

The amount of live streaming applications on IP networks has been steadily increasing with the advent of broadband technologies and novel delivery paradigms. Particularly, peer-to-peer (P2P) communications [1], which dramatically increase the number of clients a streaming session may sustain, play an important role within this trend. In P2P architectures, clients act both as receivers and senders and connect with one another as well as with servers in order to create an overlay network. For such networks, mesh topologies are preferable to trees, as they are more robust against random node departures. However, the scheduling schemes that are required in such networks are generally more sophisticated [2].

The significant path diversity of mesh overlay networks can be exploited by the concept called random linear network coding (RLNC) [3] or by its variants [4], which promises to maximize the communication rate in multicast scenarios without the need for complex scheduling schemes [5]. RLNC is performed by generating random linear combinations of the received packets at the nodes. These coded packets are then acquired by the receivers, which, in turn, decode and extract the source symbols. Since each coded packet is equally likely to be useful for decoding, RLNC provides effective data delivery without the need for a scheduling mechanism.

Although P2P networks offer scalability, bandwidth variations that arise due to the heterogeneity of the clients are quite common in such overlay networks. As a remedy, scalable video coding (SVC) concept can be used, so that the nodes in the network receive the stream with a quality that is adapted to their incoming bandwidth. For such scalable streams, RLNC can be performed in the form of prioritized network coding (PNC) [6, 7, 8, 9]. This variant of RLNC divides the source blocks into subsets that correspond to the different quality layers, where the most (least) important packets of the block are placed into the first (last) subset. Then, coding is performed across a proper collection of the first N subsets, where the resulting packet is called an N^{th}*-class* network coded packet.

In this paper, we investigate the problem of multicasting a scalable video stream in mesh overlay networks, where the objective is to maximize the average distortion reduction of the clients. The nodes perform PNC for error resiliency and enhanced throughput. We propose a pull-based (receiver-driven) scheme, where a receiver makes a decision on the numbers of the packets from each class to be requested from each of its senders. The decision process aims at maximizing the node's own expected distortion reduction, which we examine in detail. Towards this goal, receivers use local network statistics and the buffer vector messages gathered from the senders, which carry information regarding the available packets. Due to network coding, buffer vector messages consist of only the numbers of the available packets from each class, shrinking the amount of message exchange traffic. Then, the allocation problem is shown to be a log-concave optimization problem and a hill-climbing search algorithm is used to find the optimized request vectors. Simulation results show that the network resources are fully utilized in the proposed scheme, since the receiver can adjust the rate of the packets from a given class over the senders.

A few recent works have considered scalable P2P streaming algorithms with the help of network coding. In [6], a

This work has been partly supported by the Swiss National Science Foundation, under grant PZ00P2-121906.

similar approach is adopted that considers the incoming and outgoing bandwidth of the receiver nodes. However, a single request vector for all the parents is considered as a solution, which might be suboptimal. In [9], a network coding technique called the Expanding Window Random Linear Coding (EW-RLC) is employed in the proposed system, which is quite similar to PNC. The communication scenario, on the other hand, is unicast and the entire network is modeled as an erasure channel between a source and a receiver. Since the complete video stream is assumed to be available at the sender side, our proposed scheme, which assumes limited packet availability at the senders, might be considered as an extension of [9].

2. SYSTEM DESCRIPTION

We consider the decentralized distribution of a scalable video content over overlay P2P networks with mesh structures. The network is represented by a directed acyclic graph $\mathcal{G} = (V, E)$, where V and E denote the node and link sets, respectively. All the network links are lossy, where a link $(i, j) \in E$ has the packet loss probability p_{ij}. The stream is sent by servers that contain the same multimedia content that is possibly encoded differently. Rest of the nodes are heterogeneous peers that are interested in receiving the video stream with at least the basic quality and have various capabilities in terms of the upload bandwidth. The packets are sent from node i to node j with the transmission rate c_{ij}.

We assume that the video content is available as a scalable video stream consisting of L layers that correspond to different video qualities. The first layer (base layer) provides the basic quality. The other enhancement layers offer incrementally finer levels of video quality. Hence, we can provide video streams of different quality of service (QoS) to users with different resources, which is especially useful in dynamic and heterogeneous networks. The distortion reduction after successfully decoding the ℓ^{th} video layer is denoted by δ_ℓ. In a group of pictures (GOP) structure, the number of packets in the first ℓ layers is $\beta_\ell = \sum_{k=1}^{\ell} \alpha_k$, where α_k stands for the number of video packets in the k^{th} layer. Since there are L layers, the total number of source packets in a GOP is β_L.

The nodes in the network perform prioritized network coding, using the concept of generations [10]. Basically, generations are source blocks that have different decoding deadlines and consist of packets with different priorities. Coding operations are restricted to packets within the same generation. In our system, a generation corresponds to a GOP, so the generation size is equal to β_L. In particular, to create an ℓ^{th} class network coded packet, the node produces a random linear combination of at least one ℓ^{th} class packet along with lower class packets. The newly created packet belongs to the same generation as the combined packets. Arithmetic operations are carried out in a finite field $\mathbb{F}_q$. The resulting

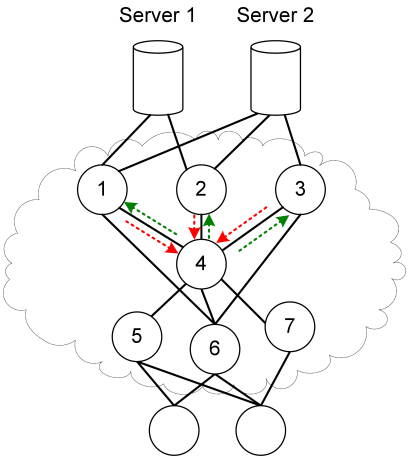

Fig. 1: Illustration of the proposed dissemination protocol. Parents are labeled as $\{1, 2, 3\}$. After receiving the buffer vectors from each parent (red), node 4 decides on the request vectors for each one of them (green). Parents send the network coded packets accordingly.

global coding vectors are appended to the packets as headers of length $\beta_L \log_2 q$ bits, along with the generation number [10]. A client decodes the first ℓ layers of a GOP as soon as it receives β_ℓ innovative network coded packets, which are all linear combinations of the source packets from the first ℓ video layers of the GOP.

3. DISSEMINATION PROTOCOL

In the proposed system, clients attempt to receive the generations successively, following their transmission order. Thus, a node is interested in receiving the generation h only after receiving the first $h-1$ generations or if the deadline of the $(h-1)^{st}$ generation has already passed. The reception of the generation h by a client i starts with the initialization phase, in which the client i receives the buffer vectors $\boldsymbol{b}_j$ from each node j in its parent set $\mathcal{P}_i$. (See Fig.1) Each of these vectors has size L, where the ℓ^{th} component $b_{j\ell}$ denotes the number of ℓ^{th} class innovative packets in node j's buffer from the generation h. The buffer vectors, along with the transmission rates and packet loss probabilities, form the information set $\mathcal{I} = \{\boldsymbol{b}_j, c_{ji}, p_{ji} \mid \forall j \in \mathcal{P}_i\}$ that is gathered and used by client i to optimize its pull decisions. Therefore, all the nodes in the overlay network periodically send their buffer vectors to their children nodes. The traffic load introduced on a link to send the buffer vectors is $L \log_2 \beta_L$ bits per generation.

The next phase in the proposed receiver-driven protocol is deciding on the amounts of network coded packets from different classes to request from the parent nodes. In other words, each node i determines the request vectors $\boldsymbol{x}_j$ to be sent to each parent j. Instead of assuming that similar sets

of packets are available at parents and therefore determining a single request vector for all of them [6], the system performance is enhanced by deciding on sepa1 request vectors for each parent. The request vector $\boldsymbol{x}_j$ has also size L and the ℓ^{th} component $x_{j\ell}$ represents the number of ℓ^{th} class packets from generation h that will be pulled down from the parent node j. The set of request vectors $\mathcal{X}_i = \{\boldsymbol{x}_j \mid \forall j \in \mathcal{P}_i\}$ is determined such that the expected video quality at node i is maximized. We know that, the distortion reduction experienced upon decoding the first ℓ video layers is $\Delta_\ell = \sum_{k=1}^{\ell} \delta_k$, provided that no layers have been decoded before. Then, the expected distortion reduction experienced by node i becomes

$$\mathbb{E}(\Delta) = \sum_{\ell=0}^{L} P_\ell \Delta_\ell, \tag{1}$$

where P_ℓ denotes the probability that node i recovers *only* the first ℓ video layers. Furthermore, we assume that the links are completely utilized, so $\sum_{\ell=1}^{L} x_{j\ell} = c_{ji}$. We can formulate the optimization problem to be solved by node i in order to maximize its expected distortion reduction as follows.

$$\max_{\mathcal{X}_i} \mathbb{E}(\Delta) \text{ s.t. } \sum_{\ell=1}^{L} x_{j\ell} = c_{ji} \text{ and } x_{j\ell} \geq 0, \ \ \forall j, \ell \tag{2}$$

Hence, client i seeks the optimal set of request vectors $\mathcal{X}_i^* = \{\boldsymbol{x}_j^* \mid \forall j \in \mathcal{P}_i\}$.

3.1. Calculation of Expected Video Quality

In this section, we investigate how the expected video quality can be calculated by node i given the information set $\mathcal{I}$. As mentioned above, $\{\boldsymbol{x}_j\}$ denotes the solution candidates for the optimum request vectors, whereas $\{\boldsymbol{b}_j\}$ and c_{ji} are buffer vectors and transmission rates that are received as feedbacks from the parents, respectively. Finally, p_{ji} can be deduced statistically by observations.

We first compute P_ℓ by considering the subspaces formed by the received packets from different classes. Let us denote by $\boldsymbol{V}_\ell$ the subspace spanned by the uncoded source packets from the first ℓ video layers, so that $\dim(\boldsymbol{V}_\ell) = \beta_\ell$. Furthermore, let $\boldsymbol{B}_{i\ell}$ denote the subspace spanned by the ℓ^{th} class innovative packets in node i's buffer, so $\dim(\boldsymbol{B}_{i\ell}) = b_{i\ell}$. To find P_ℓ, we need to consider the events in which the received packets span $\boldsymbol{V}_\ell$, but cannot span $\boldsymbol{V}_k$, for each $k > \ell$. Thus,

$$P_\ell = \Pr\left(\bigcup_{k=1}^{\ell} \boldsymbol{B}_{ik} = \boldsymbol{V}_\ell, \bigcup_{k=1}^{r} \boldsymbol{B}_{ik} \subset \boldsymbol{V}_r, r \in \{\ \ell+1, ..., L\} \right). \tag{3}$$

The first term in the joint probability of Eq.(3) can be expressed as

$$\Pr\left(\bigcup_{k=1}^{\ell} \boldsymbol{B}_{ik} = \boldsymbol{V}_\ell \right) = \Pr\left(\sum_{k=1}^{\ell} b_{ik} = \beta_\ell \right), \tag{4}$$

where $b_{ik} = \dim(\boldsymbol{B}_{ik})$ is the number of k^{th} class innovative packets in node i's buffer. Then, dropping the index i, the objective function in Eq.(1) can be rewritten as

$$\begin{aligned}
\mathbb{E}(\Delta) = \sum_{\ell=0}^{L} \Delta_\ell \sum_{r_1=0}^{\beta_1} \sum_{r_2=0}^{\beta_2 - r_1} \cdots \sum_{r_{\ell-1}=0}^{\beta_{\ell-1} - \sum_{i=1}^{\ell-2} r_i} \\
\sum_{r_{\ell+1}=0}^{\beta_{\ell+1} - \left(\sum_{i=1}^{\ell} r_i\right) - 1} \sum_{r_{\ell+2}=0}^{\beta_{\ell+2} - \left(\sum_{i=1}^{\ell+1} r_i\right) - 1} \cdots \sum_{r_L=0}^{\beta_L - \left(\sum_{i=1}^{L-1} r_i\right) - 1} \\
\Pr(b_1 = r_1) \Pr\left(b_2 = r_2 \mid b_1 = r_1\right) \ldots \\
\ldots \Pr\left(b_\ell = \beta_\ell - \sum_{i=1}^{\ell-1} r_i \mid b_k = r_k \ , \ 1 \leq k < \ell \right) \\
\ldots \Pr\left(b_L = r_L \mid b_k = r_k \ , \ 1 \leq k < L\right),
\end{aligned} \tag{5}$$

by counting all possible events and expanding the terms using conditional probabilities. Since the video layers in a practical system are not numerous, computing $\mathbb{E}(\Delta)$ does not present a challenge, since calculating the conditional probabilities in Eq.(5) is similar to the coupon collector's problem [11]. For the m^{th} term, $\Pr\left(b_m = r_m \mid b_k = r_k \ , \ 1 \leq k < m\right) \triangleq \phi_m(r_m)$, we are after the probability that node i receives r_m innovative m^{th} class packets, given it has already r_k innovative packets from each class $k < m$. However, in our case, there are a few differences. First, the packets are randomly drawn from the subspaces $\boldsymbol{B}_{jm}$ of each parent j, which means that if b_{jm} innovative packets have already been drawn, the next draws will definitely yield non-innovative packets. Therefore, the number of m^{th} class packet drawings from parent j that have a non-zero probability to be innovative, which we denote by d_{jm}, is bounded by $M_d = \min(b_{jm}, x_{jm})$. Second, d_{jm} is random due to packet erasures. Then, $\Pr(d_{jm} = d)$ (denoted by $\pi_{jm}(d)$) is given as

$$\pi_{jm}(d) = \begin{cases} \binom{x_{jm}}{d} (p_{ji})^{x_{jm}-d} (1-p_{ji})^d & \text{if } d < M_d \\ 1 - \sum_{i=0}^{M_d - 1} \binom{x_{jm}}{i} p_{ji}^{x_{jm}-i} (1-p_{ji})^i & \text{if } d = M_d \end{cases} \tag{6}$$

Since the links are independent, the probability distribution of the total number d_m^{tot} of m^{th} class packet drawings from all the parents can be found by convolving π_{jm}'s in Eq.(6) over the parents.

Finally, we need to find the probability to receive n innovative m^{th} class packets given $b_k (1 \leq k < m)$ and the total number of drawings d_m^{tot}. Obviously, the maximum number of innovative packets node i can receive is $M_{in} = \min(d_m^{tot}, \beta_m - \sum_{k=1}^{m-1} b_k)$. So we have

$$\phi_m(n) \geq \begin{cases} \binom{d_m^{tot}}{n} \gamma^n (1-\gamma)^{d_m^{tot}-n} & \text{if } n < M_{in} \\ 1 - \sum_{i=0}^{M_{in}-1} \binom{d_m^{tot}}{i} \gamma^i (1-\gamma)^{d_m^{tot}-i} & \text{if } n = M_{in} \end{cases} \tag{7}$$

where $\gamma = 1 - 1/q$ denotes the lower bound of the probability to be innovative, assuming that the linear network coding operations are performed in $\mathbb{F}_q$. Hereafter, we assume, for the sake of computational simplicity, that the inequality in Eq.(7) is actually an equality and develop the packet selection algorithm accordingly. We note that even with this assumption, the expectation of $\phi_m(n)$ is γM_{in}, which can be made arbitrarily close to M_{in} for sufficiently large values of q. Thus, in practice, the performance loss remains quite insignificant.

3.2. Packet Selection Algorithm

Since the search space is finite, a global maximum is guaranteed to exist. However, since the size of the solution space is $\mathcal{O}(\Pi_j c_{ji}^L)$, that is, it grows exponentially with the number of parents and video layers, exhaustive search algorithms are computationally expensive. Instead, we adopt a greedy hill-climbing algorithm that seeks the optimal set $\mathcal{X}_i^*$ of request vectors. To receive higher quality video, a node should allocate more upper class packets in its request vectors. However, doing so might decrease the expected distortion reduction if the lower class packets are replaced with upper class packets that are not enough to decode the upper layer and therefore useless. Alternatively, if the probability to decode an upper class ℓ is already non-zero, allocating more packets on class ℓ might increase the expected distortion reduction. We call such regions of the search space the success zones. At this point, we propose using a hill-climbing algorithm in success zones. This is because the non-zero probability to decode the ℓ^{th} layer is the summation of the binomial terms in Eq.(7), which is log-concave and therefore quasi-concave.

The algorithm runs as follows. The search space is divided into L partitions that correspond to L video layers. ℓ^{th} partition contains solution candidates with request vectors which have all zeros except for the first ℓ components. Therefore, these solutions allow the node i to decode only the first ℓ video layers. The algorithm starts with the first partition and traverses the partitions in ascending order. For each partition, a steepest-ascent hill climbing method is employed to search for a local maximum by examining the neighbors of the solution candidates. Here, we consider two sets of valid request vectors as neighbors if and only if they differ by two components of a single vector. To clarify, let us assume that all the vectors in $\mathcal{X}_i$ and $\mathcal{X}_i'$ are the same, except for $\boldsymbol{x}_j \in \mathcal{X}_i$ and $\boldsymbol{x}_j' \in \mathcal{X}_i'$. If we have $x'_{jn} = x_{jn} + 1$ and $x'_{jm} = x_{jm} - 1$ for some (n, m) pair, then $\mathcal{X}_i$ and $\mathcal{X}_i'$ are said to be neighbors. The pivotal points are selected from the edge of the success zones. At each step of the greedy search, all the neighbors of the current set of request vectors $\mathcal{X}_i$ are examined. After that, the neighbor that yields the highest $\mathbb{E}(\Delta)$ value is selected as the new local maximum candidate for the current partition. If no neighbors are able to increase $\mathbb{E}(\Delta)$, the algorithm terminates for the current partition and proceeds to the next. After each partition is checked, the local maxima that have been found are compared and the one with the highest value is the solution. This process is summarized in the pseudocode of Algorithm 1.

Except for the flat regions of a success zone in a partition ℓ, where all the packets from different classes are equally useful to decode the ℓ^{th} video layer, the algorithm performs optimally, due to the hill-climbing on a quasi-concave quantity. In addition, within such regions, the difference between the optimal and presented solutions becomes negligible for realistic parameter values, as the probability to lose all the ℓ^{th} layer packets is quite small.

Algorithm 1 Packet Selection Algorithm

for $\ell = 1$ to L **do**
  for all $j \in \mathcal{P}_i$ **do**
    if $b_{j\ell} \geq c_{ji}$ **then**
      $x_{j\ell} \leftarrow c_{ji}$ and $x_{jk} \leftarrow 0, \forall k \neq \ell$
    else
      $x_{j\ell} \leftarrow b_{j\ell}$ and distribute the remaining $c_{ji} - b_{j\ell}$ packets equally among $x_{jk}, \forall k < \ell$
    end if
    $\mathcal{X}_{best} \leftarrow \{\boldsymbol{x}_j \mid \forall j \in \mathcal{P}_i\}$, $Q_{best} \leftarrow$ vidQual$(\mathcal{X}_{best})$
  end for{Now, the pivotal point is determined.}
  repeat
    $\mathcal{X}_{current} \leftarrow \mathcal{X}_{best}$, $Q_{current} \leftarrow Q_{best}$
    for all $\mathcal{X} \in$ neighbors$(\mathcal{X}_{current})$ **do**
      if vidQual$(\mathcal{X}) >$ vidQual$(\mathcal{X}_{current})$ **then**
        $\mathcal{X}_{best} \leftarrow \mathcal{X}$, $Q_{best} \leftarrow$ vidQual$(\mathcal{X})$
      end if
    end for
  until $\mathcal{X}_{best} = \mathcal{X}_{current}$
  $\mathcal{X}^*(\ell) \leftarrow \mathcal{X}_{best}$, $Q^*(\ell) \leftarrow Q_{best}$
end for
$\mathcal{X}_{opt} = \mathcal{X}^*\big(\arg\max_\ell [Q^*(\ell)]\big)$

4. SIMULATION RESULTS

We investigate the performance of the proposed dissemination mechanism using the Foreman CIF sequence encoded by the JM8.5 of the H.264/SVC standard over random overlay networks. This video sequence has three quality layers, where the number of packets from different classes in a GOP structure are $\alpha_1 = 38$, $\alpha_2 = 15$ and $\alpha_3 = 20$. The packet size is 1500 bytes, including all the protocol and network coding headers. The GOP size is 30 frames with the frame rate 30 fps. All operations are performed in $\mathbb{F}_{2^8}$. Random mesh overlays are created by adding the peers successively to the existing network as follows. When a peer joins the network, D_{in} parents are randomly selected out of the set of nodes that have less than D_{out} children. Each node i has a specific upload bandwidth U_i that is equally distributed among its children. The transmission rate c_{ji} is then defined as the bandwidth in

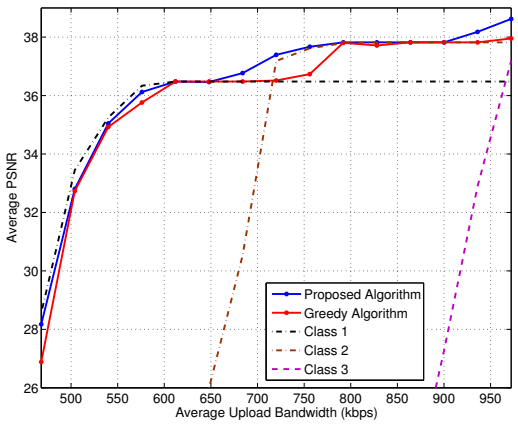

Fig. 2: Comparison of the average PSNR (dB) curves for different values of average upload capacity in the network

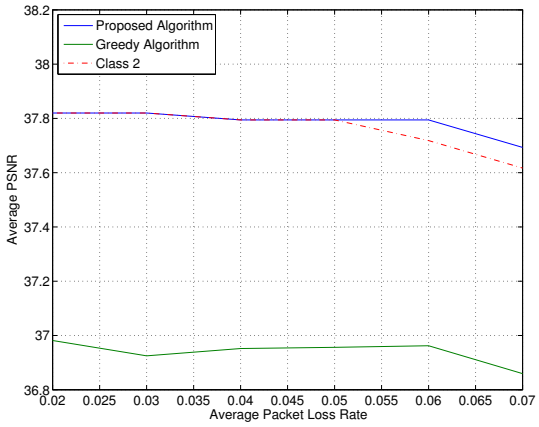

Fig. 3: Average PSNR in the network for different values of average packet loss probabilities in the network

packets per unit time allocated to the node i by node j. In our simulations, we set $D_{in} = D_{out} = 3$.

We now analyze the performance of our algorithm in terms of the average Peak-Signal-to-Noise Ratio (PSNR) achieved in the network and compare it again with [6], as well as three simple RLNC schemes where only the first, second and third class packets are randomly forwarded[1]. The results in Fig. 2 are obtained as we vary the average upload capacity of the nodes from 468 kbps to 972 kbps in a toy network with 3 servers and 18 peers. Packet loss rate is the same for all links and equals 0.05. We observe that for small values of upload bandwidth, the proposed, greedy and class 1 schemes perform equally well. As the upload bandwidths increase, the proposed scheme ensures that more nodes reach the next layer, while the other schemes require more resources for such an improvement. We can see that the proposed protocol gives better results compared to the rest when the average upload bandwidth is just sufficient to make a transition to the next video layer.

Next, for the above network construction parameters, we analyze a case where the average packet loss probability of the links is varied from 0.02 to 0.08. The average upload capacity in the network equals 745 kbps. As we see in Fig. 3, the proposed algorithm achieves a better average PSNR in the network for small packet loss rates. As the average packet loss rate increases, a growing number of nodes become unable to receive the higher quality layer. However, the rest of the nodes can still receive the higher quality by adjusting their request vectors using their parents' buffer vectors. For even greater values of the packet loss rate, almost all the nodes request only the lower class packets, which leads to a sharp decrease in the average PSNR. The greedy algorithm performs worse even for small packet loss rates, as it cannot adjust to the transmission rate variations.

We then analyze the process locally from a receiver's point of view. Starting from $t = 0$, we make significant changes to its links in terms of the packet loss rates, the upload bandwidth allocated to it as well as the packets in the buffers of its parents. The receiver has 3 parents, as visualized as in Fig. 1, which we label by $\{1, 2, 3\}$. We start at $t = 0$. First, we gradually increase the number of the 2nd class packets that parent 3 has until $t = 20$ sec. We observe that the receiver stays indifferent to the change until there is a non-zero probability to receive the 2^{nd} video layer. Then, it begins utilizing the 3^{rd} link to receive only the 2^{nd} class packets and as a result manages to decode the 2^{nd} layer. We see that it cannot adapt to this change if the greedy algorithm is employed. Subsequently, at $t = 25$ sec, we begin to increase the packet loss rate in link 1. This leads the receiver to switch back to requesting 1^{st} class packets from the 3^{rd} parent, as the 1^{st} link becomes very unreliable. The resulting drop in the video quality is due to the decrease in the effective incoming bandwidth. Finally, at $t = 40$ sec, we assume an increase in the number of the 3^{rd} class packets that the 2^{nd} parent possess, followed by a corresponding increase in the upload bandwidth it allocates to the receiver. This might happen due to random node departures. In this case, we observe that the receiver stays again indifferent at the beginning, as there is not enough effective bandwidth to receive the 2^{nd} layer. However, as both the bandwidth allocated and the number of 3^{rd} class packets in the parents increase, it is able to receive the full quality video eventually. We compare the performance of the proposed algorithm in the presented scenario to the one in [6], where we can see in Fig. 5 that the latter is outperformed; it is unable to adapt to the specific changes that the links undergo individually.

[1] In the following plots, these algorithms are dubbed as *Greedy, Class 1, Class 2 and Class 3*, respectively.

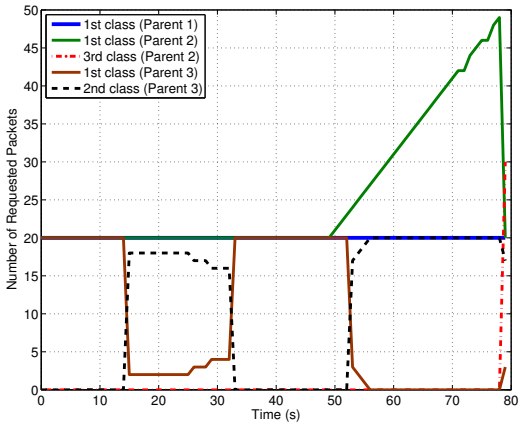

Fig. 4: Evolutions of the requested packet numbers from various classes for all the parents. The number of packets from classes that are omitted is zero for all t.

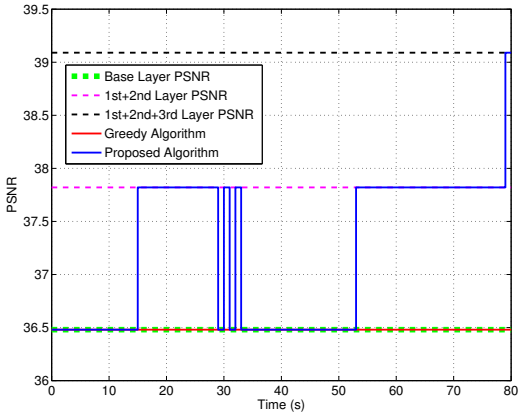

Fig. 5: Evolution of the PSNR observed by the node in the given scenario due to network dynamics.

5. CONCLUSIONS

In this paper, we have presented a novel pull-based video dissemination protocol that employs prioritized network coding. In the proposed scheme, the clients in the network request packets of different priority classes from their parents, using the information about the availability of packets sent by the parents. The nodes perform random linear network coding to minimize the code construction complexity. Unequal error protection is achieved by means of a distributed rate allocation employed by the receivers among the different classes for each parent node. As a result, the clients are able to fully utilize the available resources in an unreliable and heterogeneous network. Then, we have demonstrated the performance of the proposed protocol through simulation results, which show that our scheme is able to provide for the nodes a video quality as high as the network resources would allow.

6. REFERENCES

[1] S. Xie, B. Li, G. Y. Keung, and X. Zhang, "Coolstreaming: Design, Theory and Practice," *IEEE Transactions on Multimedia*, vol. 9, no. 8, pp. 1661–1671, Dec. 2007.

[2] Y. Liu, Y. Guo, and C. Liang, "A Survey on Peer-to-Peer Video Streaming Systems," *Peer-to-peer Networking and Applications*, vol. 1, no. 1, pp. 18–28, 2008.

[3] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, S. Jun, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. on Information Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[4] N. Thomos and P. Frossard, "Network Coding of Rateless Video in Streaming Overlays," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 12, pp. 1834–1847, Dec. 2010.

[5] M. Wang and B. Li, "Network coding in live peer-to-peer streaming," *IEEE Transactions on Multimedia*, vol. 9, no. 8, pp. 1554 – 1567, Dec. 2007.

[6] N. Thomos, J. Chakareski, and P. Frossard, "Randomized Network Coding for UEP Video Delivery in Overlay Networks," in *International Conference on Multimedia and Expo*, Cancun, Mexico, June 2009.

[7] Y. Lin, B. Liang, and B. Li, "Priority Random Linear Codes in Distributed Storage Systems," *IEEE Transactions on Parallel and Distributed Systems*, pp. 1653–1667, Nov. 2009.

[8] X. Liu, G. Cheung, and C. N. Chuah, "Structured Network Coding and Cooperative Local Peer-to-peer Repair for MBMS Video Streaming," in *IEEE International Workshop on Multimedia Signal Processing*, Cairns, Queensland, Australia, Oct. 2008.

[9] D. Vukobratović and V. Stanković, "Unequal Error Protection Random Linear Coding for Multimedia Communications," in *IEEE International Workshop on Multimedia Signal Processing*, 2010, pp. 280–285.

[10] P. A. Chou, Y. Wu, and K. Jain, "Practical Network Coding," in *Proc. 41st Allerton Conference on Communication Control and Computing*, Monticell, IL, USA, Oct. 2003.

[11] C. Fragouli and E. Soljanin, "Network Coding Applications," *Foundations and Trends® in Networking*, vol. 2, no. 2, pp. 135–269, 2007.