Perceptual quality in P2P multi-source video streaming policies

Héctor Cancela*, Pablo Rodríguez-Bocca* and Gerardo Rubino†
*InCo – Universidad de la República, Montevideo, Uruguay, e-mail: {cancela, prbocca}@fing.edu.uy
†Inria/Irisa, Rennes, France, e-mail: rubino@irisa.fr

***Abstract*— This paper explores a key aspect of the problem of sending real-time video over the Internet using a P2P architecture. The main difficulty with such a system is the high dynamics of the P2P topology, because of the frequent moves of the nodes leaving and entering the network. We consider a multi-source approach where the stream is decomposed into several flows sent by different peers to each client. Using the recently proposed PSQA technology for evaluating automatically and accurately the perceived quality at the client side, the paper focuses on the consequences of the way the stream is decomposed on the resulting quality. Our main contribution is to provide a global methodology that can be used to design such a system, illustrated by looking at three extreme cases. Our approach allows to do the design by addressing the ultimate target, the perceived quality (or Quality of Experience), instead of the standard but indirect metrics such as loss rates, delays, reliability, etc. We also propose an improved version of PSQA obtained by considering the video sequences at frame-level, instead of the packet-level approach of previous works.**
Keywords: Multi-source streaming, path diversity, quality-of-service, perceptual quality, P2P, QoE, PSQA.

I. Introduction

The nowadays increasing growth of multimedia systems present in the Internet is a consequence of the development of broadband accesses in residential users, together with the opening of content producers to new business models. It has been observed that, roughly speaking, the content's volume doubles every year, while the demand is increased by a factor of three (see `http://www.researchandmarkets.com`). These systems have many different architectures, depending on their sizes and on the popularity of their contents. The majority of them have a traditional CDN (Content Delivery Network) structure [1], [2], where a set of datacenters absorbs all the load, that is, concentrates the task of distributing the content to the customers. This is, for instance, the case of msnTV, YouTube, Jumptv, etc., all working with video content.

Another method becoming popular these days consists of using the often idle capacity of the clients to share the video distribution load, through the present mature Peer to Peer (P2P) systems. These are virtual networks developed at the application level over the Internet infrastructure. The nodes in the network, called peers, offer their resources (bandwidth, processing power, storing capacity) to the other nodes, basically because they all share common interests. As a consequence, as the number of customers increases, the same happens with the global resources of the network.

P2P networks are becoming more and more popular today (they already generate most of the traffic in the Internet). For instance, P2P systems are very used for file sharing and distribution; some known examples are Bittorrent, KaZaA, eMule, etc. Their main technical problem is that peers connect and disconnect with high frequencies, in an autonomous and completely asynchronous way. This means that the resources of the network as a whole are also highly variable, and thus, that the network must be robust face to these fluctuations.

In this paper, we are interested in some general principles that can be followed when designing a P2P architecture to distribute live video, which looks like a good idea due to the high requirements in terms of bandwidth of these applications. Streaming services in VoD (Video on Demand) have similar characteristics. However, real-time video streaming (live TV) has different and strong constraints that imply a series of specific technical problems because of the before-mentioned P2P dynamics. The main problem is how to provide good quality levels in a context where this quality depends on other clients that are delivering the stream, and given the fact that users connect and disconnect very frequently. The main idea that has been considered to deal with these problems is to build the systems using some redundancy in the signals. In this paper we explore one of them: multi-source streaming. This means that the live video stream is received by the client from flows sent by many sources simultaneously. This approach allows for a large flexibility of the system, modulated by the dynamics of the network. In particular, it is in principle possible to increase the number of sources and/or the amount of redundant information sent through the network; this opportunity can be used as a tool to deal with the problem of nodes leaving the network (we will refer to this situation as a node failure) and causing partial signal losses to some clients.

This flexibility must be carefully tuned in order to get a satisfactory quality level with a minimal cost. The usual approach is to use some metric that is known to play an important role in quality, such as the loss rate of packets, or of frames. In this paper we instead address the problem of measuring *perceived* quality by means of the PSQA technology [3]–[5]. PSQA is a general procedure that allows the automatic measurement of the perceived quality, accurately and in real-time. We extend the technique to the case of multi-source streaming for live video, and improve its efficiency for video analysis by studying the flows at the frame level, instead of the packet level previously considered in the literature.

In order to face the high dynamicity of such a system, we explore a multi-path approach where (i) the stream is decomposed in some way into several flows, (ii) each client receives those flows following different paths and sent from different other clients, (iii) the client is able to reconstruct the stream from the whole set of received flows and possibly from part of them; moreover, (iv) the system measures automatically the perceived quality at the client continuously, and takes its decisions (basically, periodically rebuilds the architecture of the network) using these values. The paper focuses then on the analysis of the impact on the perceived quality, as captured by the PSQA metric, of the fact that the stream is received from several nodes decomposed into different flows (explaining the term *multi-sourcing*). Our main goal is the description of a global methodology that can be used to design such a P2P distribution algorithm. This is illustrated by considering the extreme cases where the flows are just copies of the original sequence (a very high redundancy level) and where the sequence is split into complete disjoint sub-streams, a case where there is no redundancy at all. After some modeling work needed for the development of a PSQA module able to compute the perceived quality in real-time, we do some experiments in order to explore the consequences of these architecture choices on the quality level.

The paper is organized as follows. Section II introduces multi-source streaming techniques. Different video quality measurement techniques are presented in Section III, with particular emphasis on PSQA. Section IV develops some models needed for the construction of the PSQA measuring module. In Section V our first experimental results are introduced. The main contributions of this work are then summarized in Section VI.

II. Multi-source Streaming

The main architecture we are considering in this paper is the following one. Some server producing a live video stream splits this stream into K flows, with some amount of redundancy in them (that is, together they can transport "more information" than contained in the original video signal), and it sends each of these flows to a specific set of clients. The clients in turn send the received flows to other nodes. The architecture must ensure that each client receives the different flows from different nodes. So, from the client's point of view, we have a multi-source delivering system. Since K is a parameter of the system, the simplest situation is when there is a single server node which sends all the streaming information to the clients.

Let us consider instead the case where the server will send more than one flow composing the original signal. The quality of service perceived at the client node will be a function of the policy being used to distribute the streaming among the different flows, of the degree of redundancy, and of the loss rates and loss bursts due to transport network conditions or to instabilities at the P2P server nodes. An important complementary aspect is the degree of redundancy being employed; in this case of multiple servers, the extreme schemes are to apply no redundancy at all, or to completely replicate all the information. In the first case, we have a "split" policy: each server sends a fraction of the streaming information, without any redundancy, and the loss of information at any of these flows will imply also losses at the client. Fig. 1 represents this scheme.

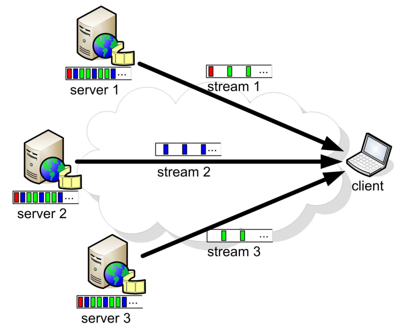

Fig. 1. Multi-source streaming split method

More precisely, we will just consider the case of sending frame 1 in flow 1, frame 2 in flow 2, up to frame K in flow K, then frame $K+1$ in flow 1, etc. (see below). In the second case, the policy being applied is "copy": each of the server nodes sends the full streaming to the client, which will then be less prone to quality problems caused by frames lost by communication problems. That is, this is the full redundant scheme where the client receives many copies of the complete flow (Fig. 2).

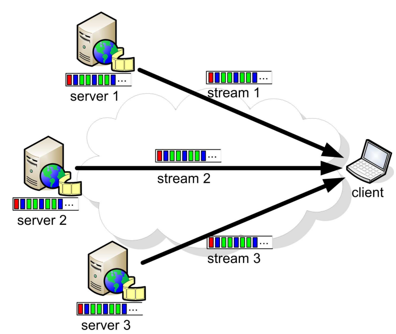

Fig. 2. Multi-source streaming copy method

Although in this work we concentrate on these extreme policies (either zero redundancy or full replication of the information sent by each server), it is clear that the degree of redundancy admits many other possibilities in-between.

In Section IV we develop models for the single server, the K-server version with split, and the K-server variant with copy configurations. The goal is to evaluate the perceived quality associated with these extreme situations. The quantitative evaluation of these models not only can give some insights into QoS characteristics of multi-source streaming in a P2P network, but also can serve as bounds for the expected behavior of other policies with an intermediate replication level.

III. Quality Measurements

This section discusses different ways of dealing with the perceived quality in a video delivering system.

A. Subjective tests

Perceived video quality is, by definition, a subjective concept. The mechanism used for its assessment is called *subjective testing*. It consists of building a panel with real human subjects, which will evaluate a series of short video sequences according to their own personal notion of quality. An alternative is to use a (smaller) panel of experts. In the first case, we will get the quality of the sequences as seen by an average observer. In the second case, we can have a more pessimistic (or optimistic, if useful) evaluation. The output of these tests is typically given as a Mean Opinion Score (MOS). Obviously, these tests are very time-consuming and expensive in manpower, which makes them hard to repeat often. And, of course, they cannot be a part of an automatic process (for example, for analyzing a live streaming in real time, for controlling purposes). There exist standard methods for conducting subjective video quality evaluations, such as the ITU-R BT.500-11 [6]. Some variants included in the standard are DSIS, DSCQS, SS, SSCQE, SCACJ and SDSCE.

B. Objective tests

Other solutions, called *objective tests*, have been proposed. Objective tests are algorithms and formulas that measure, in a certain way, the quality of a stream. The most commonly used objective measures for video are PSNR, ITS' VQM [7], EPFL's MPQM, CMPQM [8], and NVFM [8]. With some exceptions, the objetive metrics propose different ways of comparing the received sample with the original one, typically by computing some kind of distance between both signals. So, it is not possible to use them in an real-time passive test enviroment, because the received and the original video are needed at the same time in the same place. Besides, these quality metrics often provide assessments that do not correlate well with human perception, and thus their use as a replacement of subjective tests is limited.

C. Pseudo Subjective Quality Assessment (PSQA)

The Pseudo Subjective Quality Assessment (PSQA) [9] is a technique allowing to approximate the value obtained from a subjective test but automatically. The idea is to have several distorted samples evaluated subjectively, and then to use the results of this evaluation to train a specific learning tool (in PSQA the best results come from the Random Neural Networks one [10]) in order to capture the relation between the parameters that cause the distortion and the perceived quality. This method produces good evaluations for a wide range variation of all the quality affecting parameters.

Let us briefly describe the way PSQA works. We start by choosing the parameters we think will have an impact on quality. This depends on the application considered, the type of network, etc. Then, we must build a testbed allowing us to send a video sequence while freely controlling simultaneously the whole set of chosen parameters. We then choose some representative video sequences (again, depending on the type of network and application), and we send them using the testbed, by changing the values of the different parameter values. We obtain many copies of each original sequence, each associated with a combination of values for the parameters, obviously with variable quality. The received sequences are then evaluated by a panel of human observers. At that stage enters the training process, which learns the mapping from the values of the set of parameters into quality. The output of this learning process is then a function able to build a quality value from the values of the parameters.

After training, using PSQA is very easy: we need to evaluate the values of the chosen parameters at time t, and then we use the obtained function which gives the *instantaneous* perceived quality at t.

In this work, we focus on two specific parameters concerning losses, because we know from previous work on PSQA that the loss process is the most important global factor for quality. We consider the loss rates of video frames, denoted by LR, and the mean size of loss bursts, MLBS, that is, the average length of a sequence of consecutive lost frames not contained in a longer such sequence. The MLBS parameters capture the way losses are distributed in the flow. It is important to observe that in previous work using the PSQA technology the analysis was done at the packet level. Here, we are looking at a finer scale, the frame one, because quality is more directly influenced by lost frames than by lost packets. Packet-level parameters are easier to handle (in the testbed and from the measuring point of view in the network), but frame-level ones provide a more accurate view of perceived quality.

IV. Multi-source Streaming Models

In this section we develop stochastic (Markovian) models for the frame loss process in multi-source streaming, in the three considered cases, that is, for a single source, for the split case, and for the copy case.

We start from the single source case, using one of the simplest models considered in the literature, which nevertheless can take into account the two parameters of the loss process discussed in the previous section, namely the loss rate (LR) and the mean loss burst size (MLBS). We do not differentiate among losses due to the server node itself and losses due to the underlying Internet connection between server and client node; we just apply a descriptive model, whose parameters can be completely characterized by observed values of the above mentioned LR and MLBS values.

A. The simplified Gilbert model

To model the loss process on an end-to-end communication we build a discrete time stochastic process $(X_1, X_2, \ldots)$ where $X_n = 1$ if the nth frame is correctly transmitted, 0 if it is lost. The simplest i.i.d. case (a Bernoulli process) is obviously too simple because in general losses are correlated. To keep the model as simple as possible (and specially, to keep the number of parameters as small as possible) we used the so-called simplified Gilbert model, following [9], [11] (we use it at the frame level, while the original model has been proposed for packet losses, but the procedure is the same). It consists of using a 2-state Markov chain for controlling which frames are lost in the flow (so, with 2 parameters; the original Gilbert model has 3 parameters [12]). Let us denote by 1 and 0 the states, with the following semantics: after a correct transmission, we will always be at state 1, and after a loss, at state 0. Figure 3 illustrates the dynamics of the chain; at the left, the meaning of transitions, and at the right, the chain itself. The two parameters are then $p = \Pr(\text{a loss after a correct transmission})$ and $q = \Pr(\text{a correct transmission after a loss})$. In [13]–[15] this model is shown to give a good approximation of losses on the Internet (in those papers, packet losses are considered).

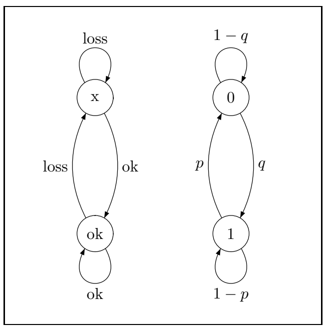

Fig. 3. The Gilbert-like model to represent the loss process and the associated 2-states Markov chain. When in state “ok”, a transition to state “x” corresponds to a loss, and to the same state “ok” it corresponds to a correct transmission. From state “x”, the loop corresponds to a loss and the transition to “ok” to a correct transmission.

The steady-state distribution of this model is given by $\pi_1 = q(p+q)^{-1}$, $\pi_0 = p(p+q)^{-1}$. The distribution of the length S of a generic burst of losses, considering the system in equilibrium, is geometric: $\Pr(S = n) = (1-q)^{n-1}q$, $n \geq 1$, with mean $\mathrm{E}(S) = q^{-1}$.

The Loss Rate LR of the flow, according to this model, and the Mean Loss Burst Size $MLBS$ of the stream, are

$$LR = \frac{p}{p+q}, \quad MLBS = \mathrm{E}(S) = \frac{1}{q}.$$

B. Sending K copies of the stream

Assume K copies of the same stream travel following independent and stochastically equivalent paths to the same terminal. The loss process at any of the K streams is represented by the model previously described, with parameters p and q. It is clear that the receiver will observe the loss of a frame if all the copies of the frames are lost. If LR_K^{copy} denotes this global Loss Rate, we then have

$$LR_K^{copy} = LR^K = \left(\frac{p}{p+q}\right)^K.$$

If S_K denotes the size of a generic burst of losses, we have $\Pr(S_K = n) = \left[(1-q)^K\right]^{n-1}\left[1-(1-q)^K\right]$, giving a global Mean Loss Burst Size $MLBS_K^{copy} = \mathrm{E}(S_K)$ as follows:

$$MLBS_K^{copy} = \frac{1}{1-(1-q)^K} = [1-(1-MLBS^{-1})^K]^{-1}.$$

C. Complete split of the stream into $K \geq 2$ substreams

In the other extreme case considered in this paper, we have K substreams transporting each a frame over K in the following way: frame 1 goes through substream 1, frame 2 through substream 2, until frame K going through substream K, then frame $K+1$ through substream 1, etc. In general, frame n is sent by substream $((n-1) \mod K) + 1$.

We obviously have here, for the Loss Rate of this scheme, the same value as for the single source case: $LR_K^{split} = LR = p(p+q)^{-1}$. The evaluation of the Mean Loss Burst Size is more involved. After some algebra, we get

$$MLBS_K^{split} = 1 + \frac{p}{q},$$

for all $K \geq 2$.

V. Testing and First Results

In this section we study how the frame loss rate and frame mean loss burst size parameters affect the quality (as measured by the PSQA technique) for the single server and multiple server (copy and split) streaming policies.

The first step was to apply the PSQA technique, as explained in Subsection III-C. This involved choosing 4 MPEG2 video sequences, of about 10 seconds duration each, with sizes between 1.5 MB and 2.8 MB). For each sequence, we generated 25 different evaluation points, where each evaluation point is defined by a loss rate value chosen at random with an uniform distribution between 0.0 and 0.2, and a mean loss burst size value chosen at random with an uniform distribution between 0.0 and 10.0. For each of the evaluation points, we used a simplified Gilbert model (discussed in Section IV-A) to simulate a frame drop history which was applied to the original video sequences; in this way, we obtained 100 modified video sequences with variable quality levels. A group of five experts evaluated the sequences and the MOS for each of the copies was computed, following the ITU-R BT.500-11 [6] norm. These MOS were scaled into a quality metric in the range $[0, 1]$.

Finally, we employed the MOS value for each of the design points as inputs in order to calibrate a Random Neural Network (RNN). After trained and validated, the RNN provides a function of two variables, LR and $MLBS$, mapping them into perceived quality (on a $[0, 10]$ range). In Figure 4 we can see the obtained function. For ease of reading, we extrapolated the curve to the borders, but observe that the data are accurate and used on an internal region ($[1\%, 20\%]$ for LR, and $[1, 10]$ for the $MLBS$). We can see that quality is monotone in the two variables, and in particular increasing with the $MLBS$, meaning that humans prefer sequences where losses are concentrated over those where losses are more isolated. We have no room here to describe in detail the learning procedure; see the references given before for similar processes.

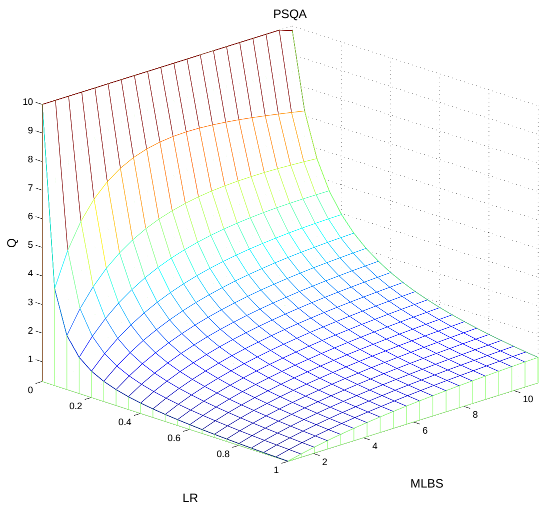

Fig. 4. The PSQA curve in our setting

With this information, we can now compare the effect of the different streaming policies on the quality perceived at the client node. In Figures 5, 6, and 7 we see respectively the situation for the single server policy, the split policy with $K = 2$, and the copy policy with $K = 2$. In the three cases we have the same frame loss process going on at the server side. The figures show the perceived quality as a function of parameter LR, for different values of $MLBS$.

In the three cases, the perceived quality deteriorates quickly with increasing values of LR; but the quality values and the shape of the curves is very different for every policy. In the single server case, the behavior is almost insensitive to the $MLBS$ parameter; the quality depends very heavily on LR, deteriorating very quickly as soon as this parameter grows a little bit from 0. In the split policy with two servers, the effect of LR is similar; we can observe that this policy completely cancels out the effect of the $MLBS$ parameter at the sources. In the copy policy, the observed quality levels for large LR are much higher than the corresponding ones in the other two models. Here the effect of $MLBS$ is more pronounced than in the previous two cases, larger $MLBS$ values having rather better quality.

In order to make it easier to compare the different policies, we show in Figures 8 and 9 the quality values for the three policies as a function of LR, with $MLBS = 1.0$ and $MLBS = 4.0$. These figures shows very clearly how the copy policy with two servers, which transmits every frame twice, profits from this redundancy and copes much better with quite high values of LR, maintaining better quality results, for both low and high $MLBS$ scenarios. In the case where $MLBS = 1$, the single and split policy give the same results (this results directly from our analytical models). When $MLBS = 4$, the split policy fares worse than the single policy. This results from the facts that in this case, the split policy results in a lower value for the $MLBS$ perceived by the user in relation to the $MLBS$ at the server side, and that the MOS computed from the experts opinions show that lower $MLBS$ result in lower quality values.

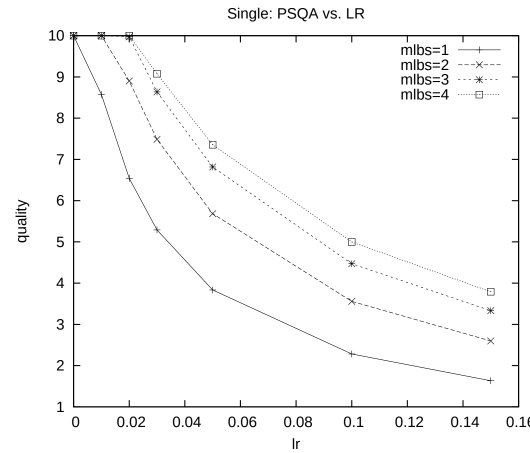

Fig. 5. Multi-source streaming Single method, in the loss domain

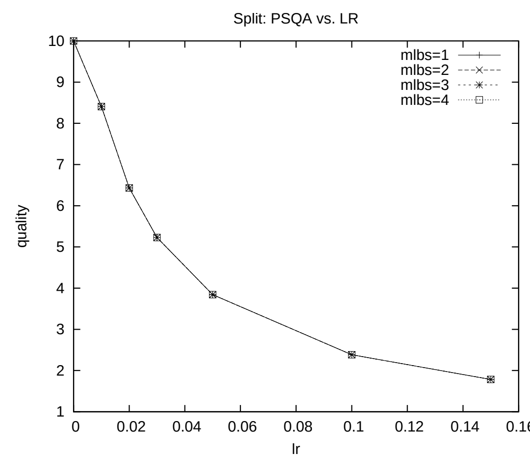

Fig. 6. Multi-source streaming Split method, in the loss domain

VI. Conclusions

This paper proposes some general principles for the design of a live-video P2P distribution system following a multi-source procedure where the video stream is decomposed into different flows that travel independently through the network. We use the PSQA technique that allows to measure automatically the perceived quality as seen by the final users. The paper focuses on the impact on quality (as measured by PSQA) on three extreme cases: sending the stream from one node to the other (for reference purposes), sending two complete copies of the stream through different paths, and sending two disjoint substreams whose union recomposes the original one.

We are able to evaluate the resulting perceived quality associated with these extreme architectures. The main conclusions are: (i) thanks to an improved version of PSQA we see that quality increases as losses concentrate in the stream (for a

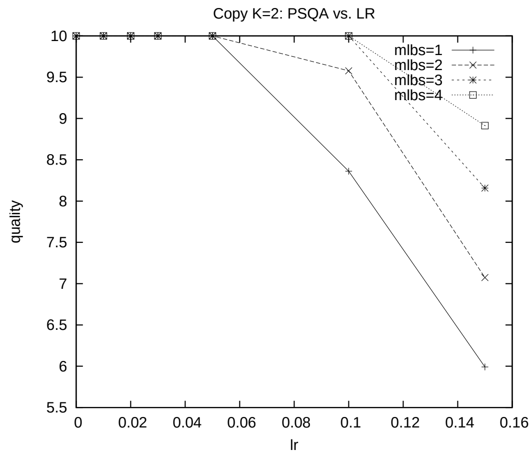

Fig. 7. Multi-source streaming Copy method, in the loss domain

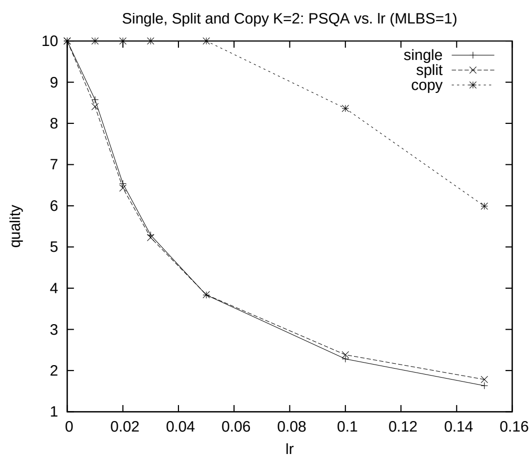

Fig. 8. Compare multi-source streamings without bursts, MLBS=1

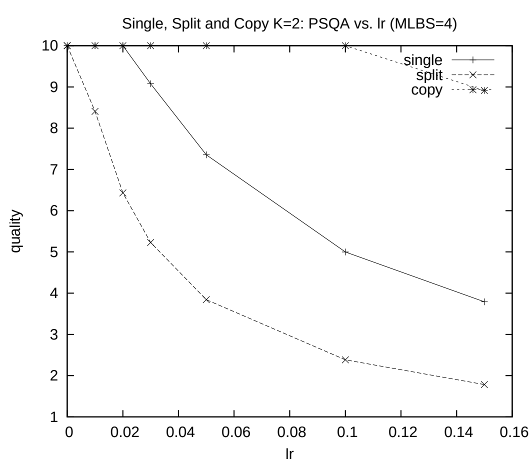

Fig. 9. Compare multi-source streamings with bursts, MLBS=4

fixed loss rate); (ii) sending the signal twice obviously leads to a much better quality, even if, as expected, losses are less concentrated in this case; (iii) sending disjoint pieces of the stream is not useful, of course under our simplifying assumptions and scenarios.

This study suggests that some intermediate point between the extreme cases we considered (and to be found in future work), perhaps with a higher number of flows per stream, can be an interesting solution. The paper proposes a global methodology that allows to look for such a scheme. It also strongly suggest to look for more realistic models, perhaps including cost aspects, to a deeper analysis of different splitting cases.

REFERENCES

[1] C. Cranor, M. Green, C. Kalmanek, D. Shur, S. Sibal, and J. V. der Merwe, "Enhanced streaming services in a content distribution network," *IEEE Internet Computing*, no. 2, pp. 66–75, 2001.

[2] S. Wee, W. Tan, J. Apostolopoulos, and S. Roy, "System design and architecture of a mobile streaming media content delivery network (msdcdn)," Streaming Media Systems Group, HP-Labs, Tech. Rep., 2003.

[3] S. Mohamed, F. Cervantes, and H. Afifi, "Audio Quality Assessment in Packet Networks: an Inter–Subjective Neural Network Model," in *Proc. of IEEE International Conference on Information Networking (ICOIN-15)*, Beppu City, Oita, Japan, Jan. 2001, pp. 579 –586.

[4] S. Mohamed, G. Rubino, H. Afifi, and F. Cervantes, "Real–time Video Quality Assessment in Packet Networks: A Neural Network Model," in *Proceedings of International Conference on Parallel and Distibuted Processing Techniques and Applications (PDPTA'01)*, Las Vegas, Nevada, USA, June 2001.

[5] S. Mohamed, G. Rubino, and M. Varela, "A method for quantitative evaluation of audio quality over packet networks and its comparison with existing techniques," in *Proceedings of the Measurement of Speech and Audio Quality in Networks workshop (MESAQIN'04)*, Prague, Czech Republic, June 2004.

[6] ITU-R Recommendation BT.500-11, "Methodology for the subjective assessment of the quality of television pictures," June 2002.

[7] S. Voran, "The Development of Objective Video Quality Measures that Emulate Human Perception," in *IEEE GLOBECOM*, 1991, pp. 1776–1781.

[8] C. van den Branden Lambrecht, "Perceptual models and architectures for video coding applications," Ph.D. dissertation, EPFL, Lausanne, Swiss, 1996.

[9] S. Mohamed and G. Rubino, "A Study of Real–time Packet Video Quality Using Random Neural Networks," *IEEE Transactions On Circuits and Systems for Video Technology*, vol. 12, no. 12, pp. 1071 –1083, Dec. 2002.

[10] E. Gelenbe, "Random Neural Networks with Negative and Positive Signals and Product Form Solution," *Neural Computation*, vol. 1, no. 4, pp. 502–511, 1989.

[11] S. Mohamed, G. Rubino, and M. Varela, "Performance evaluation of real-time speech through a packet network: a Random Neural Networks-based approach," *Performance Evaluation*, vol. 57, no. 2, pp. 141–162, May 2004.

[12] E. Gilbert, "Capacity of a burst–loss channel," *Bell Systems Technical Journal*, vol. 5, no. 39, Sept. 1960.

[13] H. Sanneck, G. Carle, and R. Koodli, "A Framework Model for Packet Loss Metrics Based on Loss Runlengths," in *Proceedings of the SPIA/ACM SIGMM Multimedia Computing and Networking Conference*, San Jose, CA, Jan. 2000, pp. 177–187.

[14] M. Yajnik, S. Moon, J. Kurose, and D. Towsley, "Measurement and Modeling of the Temporal Dependence in Packet Loss," in *Proccedings of IEEE INFOCOM '99*, 1999, pp. 345–352.

[15] J.-C. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive FEC–Based Error Control for Internet Telephony," in *Proceedings of INFOCOM '99*, New York, NY, USA, Mar. 1999, pp. 1453–1460.