TNS research in progress

Mikel Izal, Iria Prieto, Félix Espina, Daniel Morató, Eduardo Magaña

***Abstract*—This paper summarizes latest works of Telematic, Networks and Services research group (TNS) of Public University of Navarra in the topics of FIERRO thematic network. Last two papers sent to major conferences are addressed**

First work shows how self-similar traffic can be generated using Perlin Noise, an algorithm commonly used to generate 2D/3D noise for natural looking graphics. 1-dimension Perlin Noise can be interpreted as network traffic and used to generate long range dependent traffic for network simulation. The algorithm is compared to more classical approach Random Midpoint Displacement showing at traffic generated is similar but can be generated continuously with no fixed block size.

Second work presents two novel cloning schemes for video delivery in OBS networks that dramatically improve user perceived content quality. These schemes take into account the special characteristics of compressed video traffic. Analytical and simulation results show up to 40% QoE improvement without a substantial increase in the overall network traffic and without increasing the number of bursts in the network. The results show the strong dependency of this novel cloning scheme on the video traffic structure due to the coding mechanisms.

I. Work in generating FGN self similar traffic with Perlin Noise

THE fast growth of computer networks and the constant development of new services with different requirements, make necessary the design and study of different protocols and technologies through simulation. This is specially important in the case of optical next generation networks, in order to obtain results and compare performance of still-not-implemented services on network architectures. In order to simulate the behavior of these networks, different traffic models have been used in the literature.

Fractional Brownian Traffic is a self-similar traffic model which models the cumulative amount of traffic arrived, as a continuous function with gaussian increments. This increments form what is called a Fractional Gaussian Noise (FGN). FGN is characterized by three parameters. Mean and variance are the parameters of the marginal distribution of arrival process per unit time. The third parameter is the Hurst parameter (H) which measures burstiness of the process. $H = 0.5$ corresponds to the pure gaussian noise with independent arrivals (which is self-similar but not long range dependent). Larger values of $H < 1$ give increasing dependent processes being $H = 0.8 - 0.7$ typical values for Internet traces,

One Fractional Gaussian Noise, FGN, generation process is described in [1]. Due to the computational cost of the method, $O(N^2)$, several methods have been proposed after this, [2], [3]. However these methods usually generate fixed sized blocks of FGN samples, for example [1], is able to generate accurate traces with less samples than $5 \cdot 10^5$. This means that size have to be decided in advance and the full process trace has to be generated and stored before actual simulation. This limits our simulation time specially in high speed networks where we need large traces to simulate even very short times. This behavior comes from FGN generators working by generating the desired spectrum of the process and then using Fast Fourrier Transform to get the actual time-domain process or by generating farthest samples first and then refining the samples in between.

Our goal in this work is obtaining a generator to synthesize FGN on-th-fly with no need for pre-generating large blocks of samples. This generator is needed to feed simulations of high speed optical burst or packet switched networks where a large number long range dependent sources have to be provided for different inputs. To get this on-the-fly generation an adaptation of the known Perlin Noise process is proposed. Perlin Noise is usually used to generate natural looking textures and effects in computers graphics but can be reinterpreted as an FGN generator that have not been previously applied, as far as authors knowledge, to traffic generation. The accuracy of the algorithm's adaptation to obtain traces with mean, variance and Hurst parameter is compared with other FGN generators.

On the other hand FGN generates traffic with a gaussian marginal distribution and thus may generate instant peaks larger than channel capacity. The generation methodology to limit FGN traffic to channel capacity while maintaining target average and burstiness is also addressed.

II. Definitions And Notation

Perlin Noise process is defined as a sum of several noises $x_i(t)$ called octaves with increasing spectral components. Each octave is generated from an independent uniform random process. The base octave ($i = 0$) is a random independent noise generated at intervals Δt. For every $t = k\Delta t$ a random independent value is generated with a uniform distribution. For the points in between $t = k\Delta t$ the values are obtained interpolating the previous and next generated value. Interpolation can be linear or any smooth function. Successive octaves ($i = 1...n$) are built in the same way with noise taking values in an increasing number of intermediate points. In octave i the noise take new values in every $k(\Delta t/f^i)$. Each octave considered has a different (usually decreasing) amplitude A_i for the uniform random generation. Amplitude is usually given by a persistence factor p, being $A_i = p^i$.

Given n independent random noise processes $x_i(t)$ interpolated at $k\frac{\Delta t}{f^i}$. The Perlin noise is obtained as

$$n(t) = \sum_{i=0}^{n-1} p^i x_i(t) \tag{1}$$

This work was supported by the Spanish Ministry of Science and Innovation through the research project INSTINCT (TEC-2010-21178-C02-01). Also, the authors want to thank Spanish thematic network IPoTN (TEC2010-12250-E) and Public University of Navarre for funding through PIF grant

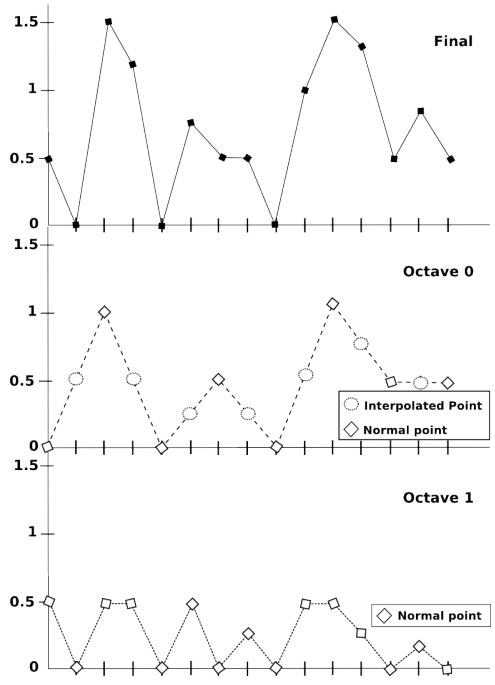

Fig. 1. Example of the Perlin Noise algorithm

Figure 1 shows an example with two octaves.

In such a process note that octave k generates f^k random values for every non-interpolated value of the first octave. In frequency domain octave k has spectral components spreading to f^k times those of first octave. In fact with this generation each octave $x_i(t)$ is approximately a white noise limited to frequency $\frac{2\pi f}{\Delta t}$.

A Perlin Noise with n octaves and interpolating factor f generates f^{n-1} samples of the highest frequency octave for each fresh sample of the first octave. For traffic generation such a process can be interpreted as providing traffic volume arriving in fixed size time slots δt given by the generation time of the high frequency octave. In that case using n octaves means that first octave generates a noise sample and moves slowly during f^{n-1} time slots to its next generated value. Therefore each octave is contributing to generate a process with correlation at least reaching $f^{n-1}\delta t$ time lags.

In this view a Perlin Noise with decaying octave amplitude is generating a long range dependent process with the decay of spectral density controlled by p. This is analogous to a Fractional Gaussian Noise with its decay of power spectral density controlled by the Hurst parameter H. From this idea the objective of this work is to generate FGN traces using Perlin Noise approach instead of RMD.

The usual interpolated factor used for Perlin Noise in graphic applications is $f = 2$. In traffic generation it is very interesting to control the reach of dependence thus larger values of f could be used to get farther correlations.

To adjust the output of Perlin Noise process notice that the amplitude of the sum process is given by 2 and the expectation can be calculated from the expectation of the uniform random generator which is a uniform noise generator $x(t)$ modulated by the octave amplitude p^i

$$Amplitude_{max} = \sum_{i=0}^{n-1} p^i = \frac{p^n - 1}{p - 1} \tag{2}$$

$$\bar{n} = \bar{x} * \sum_{i=0}^{n-1} p^i = \bar{x}\frac{p^n - 1}{p - 1} \tag{3}$$

III. Perlin Noise Traffic Generation

This section shows how to use an adapted Perlin Noise process to obtain FGN traces with the same characteristics that can be obtained using RMD method. The target FGN process is defined by mean, variance and Hurst parameter (μ, σ^2, H), given as input arguments to the algorithm. The FGN trace can be transformed to change mean and variance without changing self-similar and LRD correlation structure indicated by H. Therefore the first step to generate FGN with Perlin Noise is to get a process with desired H by choosing a persistence value p as well as the number of octaves and interpolation factor to use. The Hurst parameter depends just on p, the number of octaves n and f which define the frequency domain representation on the process.

Figure 2 show this H dependence on p, n and f. Note that for low persistence factors high values of H are obtained. This can be explained because the process obtained is a sum of components with power concentrated in increasing areas. The sum of such octaves with no modification has decaying power spectrum associated with long range dependence. To get an independent arrival process, namely $H = 0.5$, persistence has to be increased in order to give more weight to the highest octave that is already a white noise.

The number of octaves n and interpolating factor f to choose are important because the larger n and f, the larger is the number of correlated samples obtained.

As we can see in the graph, for example, in order to obtain a $H \simeq 0.7$ using 6 octaves we would have to choose a persistence near $p = 1.5$.

Therefore n octaves are generated with uniform and independent random generators with $\bar{x_i} = 0$ and weighted with p^i to obtain a Perlin Noise process $n_0(t)$. This process has desired H and mean 0.

The second step is mean and variance adjustment, obtaining a new process $n(t)$ with desired mean μ and variance σ^2.

$$n(t) = \mu + \frac{\sigma}{\sigma_0} n_0(t) \tag{4}$$

Being σ_0^2 the variance of $n_0(t)$. Note that it is easy to generate $n_0(t)$ with 0 mean but not as easy to get a normalized process with variance 1. The theoretical variance of $n_0(t)$ is derived as

$$\mathrm{V}[n_0(t)] = \sum_{i=0}^{n-1} p^{2i} V[x_i(t)] = \\ = \frac{\sum_{i=0}^{n-1} \left(2\, f^{2\,n} + f^{2\,i+2}\right) p^{2\,i}}{9\, f^{2\,n}} \tag{5}$$

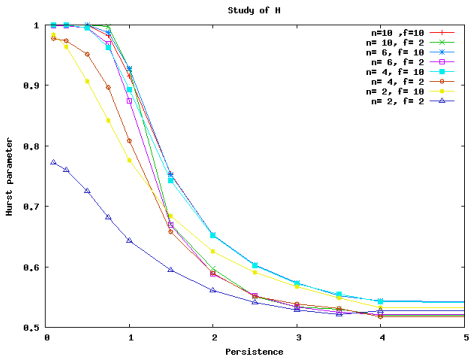

Fig. 2. H parameter depending on number of octaves, persistence and interpolation factor

After rescaling the process to get mean and variance, H parameter keeps unchanged, thus a FGN trace is obtained with target mean, variance and H and correlation reaching as far as f^{n-1}. It should be noted that the value of σ_0 could also be obtained from the actual synthesized $n_0(t)$ once it has been generated. However this work's objective is to generate continuously the process $n(t)$ with no need to generate fixed size block traces in advance. The theoretical value of σ_0 allows the generation of samples of $n(t)$ as soon as samples from $n_0(t)$ are generated.

IV. Generating limited capacity channel traffic

In the previous section a method is provided to generate a FGN with desired mean and variance $n(t)$ by scaling a Perlin Noise process $n_0(t)$. Usually this is interesting to generate traffic on a given channel with limited capacity C in order to simulate network behavior. The problem with this output process is that once mean and variance are chosen the maximum values of traffic volume per sample time may be higher than channel capacity limit or even lower than 0. The FGN process has been generated without those limits in mind.

To fix that, an algorithm is needed that shape the output $n(t)$ to keep it within limits of channel capacity and over 0. The algorithm needs to preserve the mean and as much as possible also variance and Hurst parameter (i.e. correlation structure) in the capacity limited process $n_s(t)$

The algorithm used is shown in figure 3. In order to keep the mean in the system every time generated traffic is over capacity limit, the volume exceeding the capacity is stored to be added to the traffic generated in the next interval. Therefore in periods where traffic generated is over capacity limit for extended time, the cumulative stored traffic is generated after the high traffic epoch ends. This keeps the same expectation of $\bar{n_s}(t) = \bar{n}(t)$ provided that $\bar{n}(t) < C$. On the opposite side when the generated traffic volume in an interval is negative it is also added to the traffic generated in the next interval thus reducing the generated traffic after that and keeping the mean of $n_s(t)$ equal to that of $n(t)$.

An example is shown in figure 4. It shows the bytes for one Perlin Noise trace with the parameter values of: $H \simeq 0.994$, $\mu = 0.3$ and $\sigma^2 \simeq 0.2$. It can be seen how a initial process

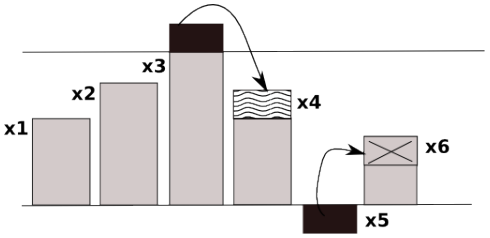

Fig. 3. Algorithm to preserve the mean for real system with capacity

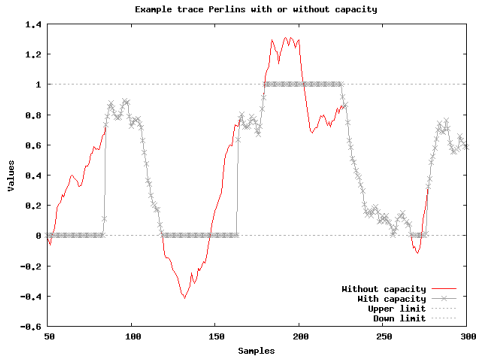

Fig. 4. How really works the algorithm to preserve the mean for real system with capacity showing the acumulative bytes of a Perlin trace

is followed by the shaped version. The shaped one has never a slope greater that the one given by capacity ($C = 1$ in this case) or lower than 0. However due to the build method it has the same overall growing rate than the original one $\mu = 0.3$.

Shaping the process to obtain desired capacity limit is easy to do with previous algorithm. The problem is the effect of that shaping in the output process variance and correlation structure given by Hurst parameter. It is clear that a process with limited range can not have arbitrary large variance. In the section that follows the limits of variance and H that can be expected from the limited capacity algorithm are shown. Without loss of generality we normalize the capacity limit as $C = 1$ and scale all parameters to fit.

A. Absolute error of Variance

Although the shaping process does not modify the mean of the output traffic process. Variance may be clearly affected. The channel limits will not allow variance to grow too high even if the original process can have arbitrary large variance. Also note the maximum variance may depend on the mean value of the generated traffic. A process with mean $\mu = 0.5$ can be easily expected to have $\sigma \simeq 0.25$. But the same value of variance seems too high for a process with $\mu = 0.01$ since values lower than 0 will be cut.

To evaluate the reasonable values of target μ, σ^2 that can be reasonably reached with the algorithm traces with different μ,σ^2 are generated and the actual variance obtained after shaping is compared to the target variance. Figure 5 shows (μ, σ) plane indicating the area where variance error is less than a given one i.e. 0.1 for $H = 0.5$. For the rest of $H > 0.5$ the variance error is even lower so the same area is safe to generate for every H value.

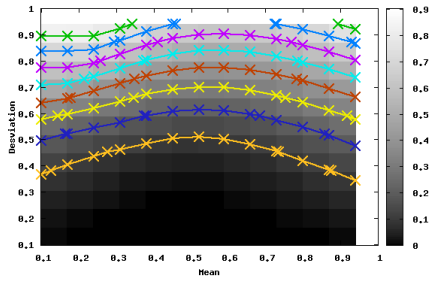

Fig. 5. Absolute error for Traces with H=0.5

V. Comparing RMD to Perlin Noise

In this section results for Perlin Noise-based generator are compared to those obtained using RMD algorithm [4]. This algorithm has been chosen since it is an approximate method known by his accuracy to obtain FGN traces in a efficient way. Maximum channel capacity is normalized to $C = 1$.

Table I show target values of μ, σ^2 and H for an example of comparison. These values are within safe zone as seen in previous section and generated trace obtain close actual values as seen on table. A slightly lower variance is expected because of limited channel capacity that reduce the range of possible values. H value is obtained with variance-aggregation estimator. For Perlin Noise generator the number of octaves used was $n = 6$ which gave a persistence value $p = 1.4$ to achieve a Hurst parameter $H \simeq 0.7$.

TABLE I
Comparative of obtained parameters through Perlin Noise and RMD generators taking into account limited capacity $C = 1$

	Target	Perlin generated	RMD generated
μ	0.380	0.380	0.381
σ^2	0.096	0.0832	0.0927
H	0.7	0.732	0.724

In order to check Perlin trace self-similar and long range dependence properties, its spectral density is compared to that of the RMD generated traces.

Spectral density for both traces is shown in figure 6. The best fit to the spectral density of a self-similar process is also plotted showing similar H values to the ones obtained with variance-aggregation estimator. Thus spectral density of the trace generated with Perlin Noise algorithm is that of a self-similar process with the target H value.

Therefore the algorithm generate traces with correct parameters comparable to the ones generated by RMD and it can be operated to generate a continuous series of values without fixed size blocks.

VI. Conclusion

Self-similar traffic models are commonly used in order to feed simulations of high speed optical burst or packet switched networks with real-like traffic. Perhaps the most used of these is the Fractional Gaussian Noise (FGN) due to its simplicity.

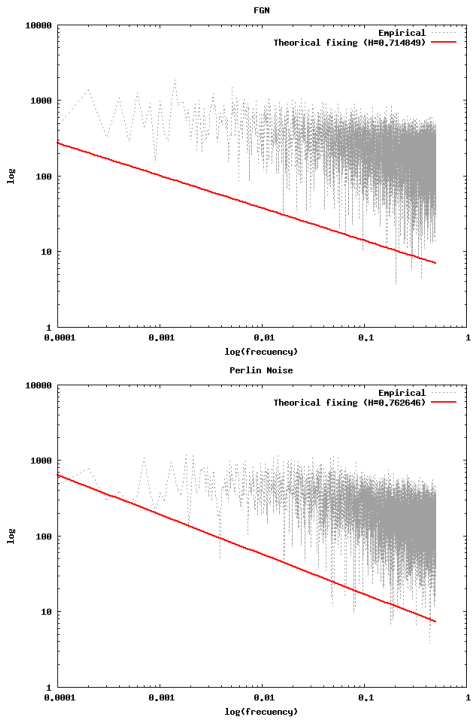

Fig. 6. Spectral density of FGN and Perlin traces with H=0.7 and approximation by straight lines

Several methods have been described in the literature to achieve a compromise between accurate and efficiency in the generation of FGN traces. However the main disadvantage of them is that trace length has to be decided in advance. In this paper it has been shown that an FGN generator can be built as an adaptation of the well known Perlin Noise process which has been often used in computer graphics. This adaptation has been reinterpreted as an FGN modeling network traffic in a fixed capacity channel.

The main advantage of this Perlin noise based generator is that it can generate traffic on-the-fly with no need to store the self-similar traffic trace in advance.

The accuracy of the generator has been compared to that of Random Midpoint Displacement algorithm, the classic FGN generator. RMD is known to be fast, simple and efficient. Perlin noise algorithm has been shown to obtain similar results generating random traffic with the same parameters than a RMD generator.

The long range dependency and spectral properties of Perlin generated traffic have been shown to be equivalent to those of a process generated with RMD and that they fit theoretical characteristics of FGN.

Therefore Perlin noise based traffic generator can be used as a long range dependent traffic source of FGN that can be generated continuously removing the need of large size trace generation before simulation.

VII. Work in Burst Cloning Techniques for Video Quality Improvement in OBS Networks

Many mechanisms have been proposed to reduce the loss rate in OBS networks: retransmission, forward error correction, cloning, contention resolution (fiber delay lines, deflection routing, wavelength conversion, segmentation), contention avoidance, etc.

Contention resolution schemes appear to be the best solution to reduce the loss rate in OBS networks, but nowadays they present some serious implementation difficulties

Retransmission schemes are not useful in OBS core networks because of the high latency introduced and the requirement of larger buffers on the edge nodes. An alternative to retransmissions is data cloning at the ingress nodes, creating copies of the input traffic that improve delivering probabilities. Almost all the burst cloning proposals clone all the traffic and hence increase network load to at least twice the original one. None of these schemes is designed to improve video quality taking into account the particular characteristics of video. Only [5] appears to propose a cloning scheme for video, but it does not perform any analytical study of the video quality improvement or the selection of frames to clone.

This work introduces two novel schemes for video delivery over OBS networks based on burst cloning and the particular characteristics of video. These schemes only clone some selected video frames. Frame selection is based on the effect that their loss would have on video quality.

VIII. Video QoE analytical model

Compressed video traffic presents a strong correlation structure, partially due to the coding algorithms that take advantage of the slow changes in the images in order to reduce frame sizes. MPEG family coders [6] have been widely adopted for video codification. Three types of video frames are defined in the MPEG standards: intra-coded frames (I-frames), inter-coded or predicted frames (P-frames) and bidirectional coded frames (B-frames). I-frames do not depend on any frame in their coding/decoding process. P-frames depend on the previous I- or P-frame. B-frames depend on the previous I- or P-frame and the following one of either type.

The set of frames between two intra-coded frames (I-frames) is named Group of Pictures (GoP). A typical nomenclature for a regular GoP structure is $GxBy$, where x is the total number of frames and y the number of consecutive B-frames. A typical GoP structure is for example $G12B2$ or *IBBPBBPBBPBB*. It is an open GoP, because the last B-frames depend on the I-frame from the next GoP. In a closed GoP there is no dependence on frames out of the GoP, like for example in $G9B3$ or *IBBBPBBBP*. The number of frames from each type in a GoP is obtained using (6).

$$\begin{aligned} G_I &= 1 \\ G_P &= \left\lfloor \frac{x-1}{y+1} \right\rfloor \\ G_B &= x - 1 - \left\lfloor \frac{x-1}{y+1} \right\rfloor = y \left\lceil \frac{x-1}{y+1} \right\rceil \end{aligned} \tag{6}$$

A GoP is an open one if $x/(y+1)$ is an integer number, and a closed one if $(x-1)/(y+1)$ is an integer number. Therefore, we can define the variable z (7), that thames the value 1 if the GoP is an open one, and 0 if it is a closed GoP:

$$z = \left\lfloor \frac{N - 1 - N_P}{(M-1)(N_P - 1)} \right\rfloor \tag{7}$$

This hierarchical structure of MPEG encoding implies a possible error propagation through its frames, and therefore it adds an extra handicap to the transport of MPEG video flows over lossy networks [7]. Frames that arrive at the destination could be useless if the other frames that they depend on have been dropped by the network, and so, small frame loss rates may cause high frame error rates, degrading the video quality perceived by the user.

A video provider needs to quantify the video quality problems, at best before the user perceives them. The concept of *Quality of Experience (QoE)* emerged from this need. ITU-T defines QoE [8] as "The overall acceptability of an application or service, as perceived subjectively by the end-user".

The user perceives the time periods when video is not correctly decoded. This can be measured as the number of seconds over one hour that could not be displayed (VT_{nd}). The proportion of frames that could be decoded, and therefore displayed, is named the *Decodable Frame Rate Q*, so $VT_{nd} = (1-Q) * 3600$. A value of $Q = 0.99$ indicates that, on average, 36 seconds of video will not be correctly decoded for each hour. An analytical model for scenarios where frame losses can be considered mutually independent was presented on [9]. The model from [9] is only valid for open GoPs, but it is easily extended for closed GoPs (8).

$$\begin{aligned} Q = &\frac{(1-P_I) + (1-P_I)\left[1 + y(1-P_B)\right] \sum_{i=1}^{G_P} (1-P_P)^i}{x} + \\ &+ \frac{zy(1-P_I)^2(1-P_B)(1-P_P)^{G_P}}{x} \end{aligned} \tag{8}$$

$P_{\{I,P,B\}}$ is the probability for a {I, P, B}-frame getting lost.

OBS burst losses can be considered mutually independent [10], so if each burst contains at most one whole video frame, video frame losses can be considered mutually independent too. Video streaming servers usually send the packets from a video frame back-to-back, so they arrive about the same time to the burstifier and it can be assumed that they are aggregated into the same burst. If the timer T_{out} is smaller than the inter-frame time T_{if} (typical value of 40ms), the video frame losses on this OBS network can be considered mutually independent and (8) can be applied.

IX. Proposed Cloning Schemes

In the Burst Cloning Scheme proposal [11], one or more cloned bursts can be created from each original burst and scheduled for transmission at the output port of the ingress node. If at least one of these bursts arrives at the destination, the original burst is considered to be successful. If more copies are made for a particular burst then it is less likely to become unsuccessful.

The major side effect of burst cloning is an increase of the network load. Each network link carries on average twice the original load or more, as some studies suggest making more than one copy for the original burst [12]. However, increasing the number of cloned bursts per original one can cause the opposite effect, a higher contention probability resulting in higher loss rates.

Two different schemes are proposed in this paper: Frame Duplication at Next Burst (*FDNB*) cloning scheme and Frame Duplication at Exclusive Burst (*FDEB*) cloning scheme. Both scheme avoid cloning all incoming traffic. FDNB duplicates into the next burst only selected types of video frames (*priority frames*). FDEB duplicates the *priority frames* creating an independent burst. The priority frames are selected such that video quality improvement is maximized, i.e. such that they minimize the VT_{nd}. To the best of the authors' knowledge, the improvement of video QoE using OBS cloning has not yet been analytically studied.

The FDEB Ingress node in Fig. 7 consists of three burstifiers per egress node. One burstifier is used for Best Effort traffic, while the other two burstifiers are used simultaneously and with a common timer, for video traffic. *Burstifier A* aggregates all the incoming video traffic and *burstifier B* only aggregates a copy of selected frames from each video. When the common timer expires, the bursts from both burstifiers are scheduled for transmission at the output port. If *burstifier B* is empty then only the burst from *burstifier A* is scheduled. Using this procedure, the FDEB duplicates the priority frames from each video using exclusive bursts for them.

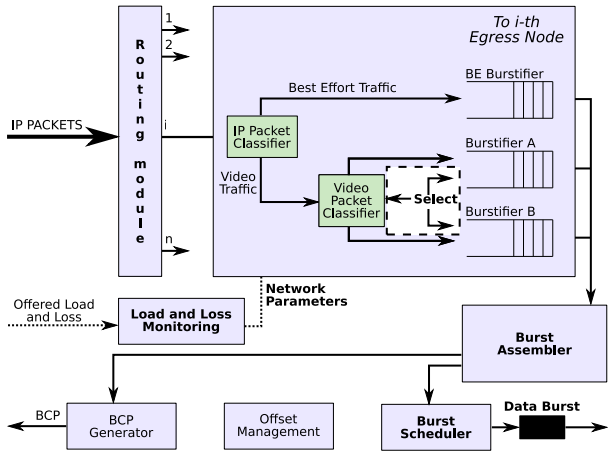

Fig. 7. Diagram of Ingress node, where dotted section applies only to FDNB

The FDNB Ingress node adds the "select" signal to the FDEB Ingress node (dotted section in Fig. 7). Every time a burst is generated, the value of the "select" signal is toggled. If the signal is "0" then burstifier *A* aggregates all the incoming video traffic while burstifier *B* aggregates only *a copy of the priority frames* for each video, thus priority video frames get duplicated. The burstifier that aggregates the total video traffic is the only one with an active formation timer. In this case with signal value "0", it is burstifier *A* the one with an active timer. When the timer expires, the burst from burstifier *A* is scheduled for transmission at the output port and the "select" signal flips to "1". Now, all the incoming video traffic will go to burstifier *B*, that stored duplicate priority frames from the previous burst. Meanwhile, the arriving priority frames get duplicated into burstifier *A*. Using this procedure, the FDNB duplicates the priority frames from each video at the next burst.

Both FDEB and FDNB increase in the same proportion the network load, because both duplicate the same priority frames but in different ways. FDEB creates extra bursts to duplicate the priority frames, while FNDB does not. As all the videos to the same egress node are multiplexed into the same *video burstifiers*, in the worst case, an extra burst per egress node is created every T_{out} seconds. This increases the number of required BCPs. But more important, as the extra bursts could be aggregating low amounts of traffic, burst sizes could result below the minimum required for OBS optical switching limits. Therefore, padding could be required for the extra bursts, increasing even more the network load. FDNB does not created extra bursts, but it does increase the burst sizes.

On both schemes, the selected priority frames could be different for each video, although they use the same *video burstifiers*. This is accomplished by the *Video Packet Classifier* module at the ingress node. The selection will depend on the loss rate incurred at the core nodes and the GoP structure of the video. When there is a high loss rate, less cloned frames are preferred to minimize its contribution to the network load. So, with high loss rate less or no priority frames will be selected from each video.

As expected, and as the simulation results will show, for both schemes the best improvement is obtained by cloning the entire video. However, in some situations this will dramatically boost the network load and the overall loss rate, seriously affecting the best effort traffic. In these cases, only some frames should be cloned, and surprisingly, the I-frames are not always the selected ones and the best choice depends on the GoP structure of the video.

X. Results

Simulations were made to measure the video QoE improvement achieved by the proposed FDEB and FDND cloning schemes for different priority frames selection policies. A specific simulator for an OBS network was developed. As a first approach, the OBS core network is modeled as a black box with a burst loss probability p not affected by the load increase from cloning. Different network scenarios were evaluated using $p = \left[10^{-4}, 10^{-2}\right]$.

The video source uses a video trace describing the size and time for each frame. Different video traces from [13] were used as video flows, but for briefness, only results for some of them are presented.

Five configurations where evaluated: without cloning; all frames are cloned; only I-frames are cloned; only P-frames are cloned; and only B-frames are cloned. The timer T_{out} used at the burstifiers is limited to a value smaller than the inter-frame time T_{if} (around 40ms), so the frame losses on the OBS network can be considered mutually independent. The frame loss probability is equal to the burst loss probability p,

except for the priority frames, where it will be the probability of losing both the burst with the original frame and the burst with the cloned one: p^2.

Fig. 8 shows the simulation and analytical results for the Decodable Frame Rate. Results are obtained at 95% confidence level, but most confidence intervals are too small to be noticed in the figure. For space constraints, only *Star Wars IV* trace is plotted. Regardless of GoP structure analytical results (8) match quite well with the simulations for traces. Cloning all the frames offers the best results in Decodable Frame Rate but at the highest cost in network traffic increase. The second best choice is obtained by I-frames cloning for the *Star Wars IV* trace show, but P-frames cloning achieves better results for other traces This means that Decodable Frame Rate reduction by cloning depends on characteristics from the video.

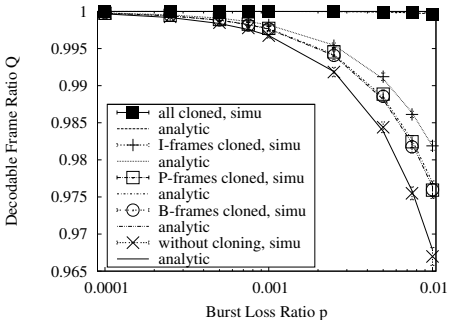

Fig. 8. Decodable Frame Rate for Matrix I and Star Wars IV traces

Fig. 9 shows the video quality (VT_{nd}) obtained analytically. As expected, all cloning configurations improve the video quality. The configuration that obtains the second best results in VT_{nd} (second after the case of cloning all the frames), varies from one trace to another. For several traces like *Matrix I*, the best choice is to clone the P-frames, but for the *Star Wars IV* trace it is to clone the I-frames. The analytical equation (8) only takes into account the GoP structure of the video. Therefore, for the same p, the GoP structure is a determinant factor for the selection of the priority frames.

The results so far did not take into account that cloning increases the network load, and therefore the output port contention and the network burst loss probability. This higher loss probability has a negative effect on video quality and could reduce the gain obtained by cloning. It could also have a high impact on the best effort traffic.

The OBS core network path can be modeled using M consecutive and independent bufferless servers. Assuming that the traffic to each OBS switch output port in the path arrives from many independent sources, the aggregate traffic to the output port can be considered as a Poisson process. Therefore, the server burst loss probability p_0 can be computed using the Erlang-B formula [14]. Coming the traffic to each switch from independent sourced then the burst loss probability in the path is just $p = 1 - (1 - p_0)^M$.

Fig. 10 shows the video quality and the network burst loss probability for the five cloning configurations. The network has four core nodes ($M = 4$) from the ingress node to the egress node and each optical link has 16 wavelengths at 1Gbps. The network carries N video flows from the ingress node to the egress node and a background traffic of medium ($\rho = 0.5$) utilization. Again, cloning all the frames outperforms the rest of the alternatives, even though it highly increases the burst loss probability. The effect on the video from this increase in the loss probability is compensated by the loss recovery obtained by cloning all the frames, but for the rest of the traffic (the best effort traffic) it represents unacceptable high loss rates, even at low utilization. In contrast, the duplication of I- or P-frames improves the video quality without significantly increasing the burst loss probability. For example, at $\rho = 0.5$ and adding 100 videos, the duplication of I-frames improves the video quality up to 40%, from $VT_{nd} = 276.25$ secs to 164.30 secs, and the duplication of P-frames improves the video quality up to 25% ($VT_{nd} = 207.18$ secs).

XI. Priority frames selection based on GoP

As shown in section X, the GoP structure is of great importance for priority frames selection. The priority frames should be I- or P-frames depending on the GoP structure of the video. Cloning B-frames does not improve in a significant amount the video quality and cloning all frames increases too much the global burst loss probability.

We define variable δ as the difference between the video quality when I-frames are cloned and the video quality when P-frames are cloned, i.e. $\delta = x * (VT_{nd,I} - VT_{nd,P})/3600$. If δ is negative, the best choice is to clone the I-frames, and if it is positive it is better to clone the P-frames. For the scenario studied in this paper, δ can be computed by (9).

$$
\begin{aligned}
\delta &= x(1-Q_I) - x(I-Q_P) = x(Q_P - Q_I) = \qquad (9)\\
&= (1-p) + [(1-p) + y(1-p)(1-p)] \sum_{i=1}^{G_P} (1-p^2)^i + \\
&+ zy(1-p)^2(1-p)(1-p^2)^{G_P} + \\
&- (1-p^2) - \left[(1-p^2) + y(1-p)(1-p^2)\right] \sum_{i=1}^{G_P} (1-p)^i + \\
&- zy(1-p^2)^2(1-p)(1-p)^{G_P} = \\
&= -p(1-p) + zy(1-p)^{G_P+3} \left[(1+p)^{G_P} - (1+p)^2\right] + \\
&+ \left[(1-p) + y(1-p)^2\right] \sum_{i=1}^{G_P} (1-p)^i \left[(1+p)^i - (1+p)\right]
\end{aligned}
$$

The first term in (9) is always negative and it does not depend on the GoP structure. The last term is always equal or greater than 0, but its value depends on the GoP structure and it decides, for closed GoPs, the priority frames to select. The second term is only for open GoPs and depending on the GoP structure is negative, positive or zero. It can be seen that the decision to choose the I-frames or the P-frames as priority frames depends on the x and y of the $GxBy$ GoP structure.

Using δ, some conclusions can be extracted. If the GoP has only one P-frame, then $\delta = -p(1+p) - zyp(1-p)^4(1+p)$ is always negative regardless of x and y. The best priority frames for GoPs with only one P-frame are always I-frames.

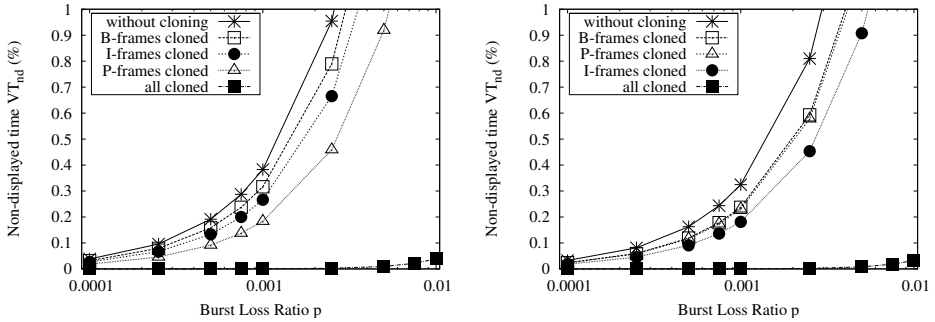

Fig. 9. Video QoE measured as video time that could not be displayed (VT_{nd}) for the different scheme configurations and traces

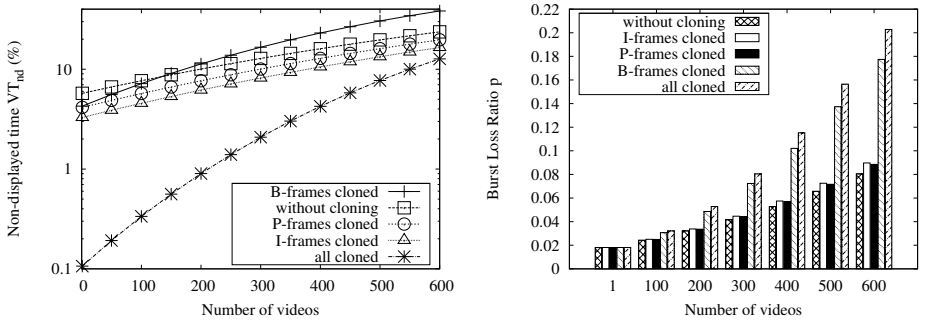

Fig. 10. Video quality (VT_{nd}) and network burst loss ratio p for the Star Wars IV traces when network load is taken into account

This analytically explains the results obtained from the trace *Star Wars IV* with $G16B7$ GoP structure.

If the GoP has exactly two P-frames, then $\delta = p(1+p)\left[(1-p)^2\left((1-p)+y(1-p)^2\right)-1\right]$ regardless of being an open or closed GoP. δ will be positive, i.e. the best priority frames will be P-frames, for any y if $p < 0.18$.

XII. Conclusions

This work presents two novel video QoE improvement schemes for OBS networks: the Frame Duplication at Exclusive Burst cloning scheme and Frame Duplication at Next Burst cloning scheme. The most important frames for video quality are cloned into an extra burst or into the next burst, respectively. The proposed ingress nodes have an additional video burstifier to make this cloning easier. The simulations and the analytical model show a significant improvement of quality, up to 40% in the simulations, in some cases with virtually no traffic increase. It was shown that the selection of frames for cloning has strong dependence on the GoP structure. A way to easily select the *priority frames* was presented as long as some selection examples.

References

[1] C. Huang, M. Devetsikiotis, I. Lambadaris, and A. Kaye, "Fast simulation for self-similar traffic in atm networks," in *Communications, 1995. ICC '95 Seattle, 'Gateway to Globalization', 1995 IEEE International Conference on*, vol. 1, jun 1995, pp. 438 –444 vol.1.

[2] D. Ostry, "Synthesis of accurate fractional gaussian noise by filtering," *Information Theory, IEEE Transactions on*, vol. 52, no. 4, pp. 1609 –1623, april 2006.

[3] E. Perrin, R. Harba, R. Jennane, and I. Iribarren, "Fast and exact synthesis for 1-d fractional brownian motion and fractional gaussian noises," *Signal Processing Letters, IEEE*, vol. 9, no. 11, pp. 382 –384, nov. 2002.

[4] W.-C. Lau, A. Erramilli, J. Wang, and W. Willinger, "Self-similar traffic generation: the random midpoint displacement algorithm and its properties," in *Communications, 1995. ICC '95 Seattle, 'Gateway to Globalization', 1995 IEEE International Conference on*, vol. 1, jun 1995, pp. 466 –472 vol.1.

[5] S. Askar, G. Zervas, D. K. Hunter, and D. Simeonidou, "A novel ingress node design for video streaming over optical burst switching networks," *Opt. Express*, vol. 19, no. 26, pp. B191–B196, Dec 2011.

[6] "Home page of the Moving Picture Experts Group (MPEG)," septembre 2011. [Online]. Available: http://mpeg.chiariglione.org/

[7] O. A. Lotfallah, M. Reisslein, and S. Panchanathan, "A framework for advanced video traces: evaluating visual quality for video transmission over lossy networks," *EURASIP J. Appl. Signal Process.*, vol. 2006, pp. 263–263, january.

[8] "ITU-T Rec. P.10/G.100 (incl. Amendment 2): Vocabulary for Performance and Quality of Service," 2008.

[9] A. Ziviani, B. E. Wolfinger, J. F. Rezende, O. C. Duarte, and S. Fdida, "Joint adoption of qos schemes for mpeg streams," *Multimedia Tools Appl.*, vol. 26, pp. 59–80, May 2005.

[10] X. Yu, J. Li, X. Cao, Y. Chen, and C. Qiao, "Traffic Statistics and Performance Evaluation in Optical Burst Switched Networks," *IEEE Journal of Lightwave Technology*, vol. 22, no. 12, pp. 2722– 2738, December 2004.

[11] X. Huang, V. Vokkarane, and J. Jue, "Burst cloning: a proactive scheme to reduce data loss in optical burst-switched networks," in *Communications, 2005. ICC 2005. 2005 IEEE International Conference on*, vol. 3, may 2005, pp. 1673 – 1677 Vol. 3.

[12] M. Gonzalez-Ortega, J. Lopez-Ardao, C. Lopez-Garcia, P. Argibay-Losada, R. Rodriguez-Rubio, and M. Pineiro-Valladares, "Loss differentiation in obs networks without wavelength-conversion capability," *Communications Letters, IEEE*, vol. 12, no. 12, pp. 903 –905, december 2008.

[13] P. Seeling, M. Reisslein, and B. Kulapala, "Network performance evaluation using frame size and quality traces of single-layer and two-layer video: A tutorial," *Communications Surveys Tutorials, IEEE*, vol. 6, no. 3, pp. 58 –78, quarter 2004.

[14] *Performance comparison of scheduling algorithms for IPTV traffic over polymorphous OBS routers*, dec. 2007.