A Gamma Fractal Noise Source Model for Variable Bit Rate Video Servers

Raúl V. Ramírez-Velarde, [1] and Ramón M. Rodríguez-Dagnino, [2]

Monterrey Institute of Technology (ITESM)
Sucursal de correos "J" C.P. 64849, Monterrey, N.L., México

ABSTRACT

In this paper, we model the statistical behavior of real traces of MPEG-4 encoded variable bit rate (VBR) video data in a digital video server. As performance measures we calculate bounds for the user disk service rate and buffer size. According to our simulations, the Gamma Fractal Noise model for the increment process, which is self-similar, seems to be very close to the actual data behavior.

1. INTRODUCTION

The proper design of a Digital Video Server System (DVSS) allows to have more users with adequate response time in a cost-effective manner. The applications of these systems include entertainment as a video rental center, interactive television, digital radio and television, remote shopping, home and office automation, and distance learning.

Basically, in these applications the user can access any video segment from the catalog as long as it is available in storage. We assume in this paper that a video segment is stored on a single disk in digital format, delivered through a high-speed telecommunications network on the frame-by-frame basis and the requested video will always be available for the subscriber.

It is desirable that the server be able to attend as many as possible simultaneous subscribers. Hence, the service time allocated to each user needs to be fine-tuned to ensure that there will be no starvation of video frames. It is also necessary to allocate an adequate buffer space for each user video strand, as seen in Figure 1.

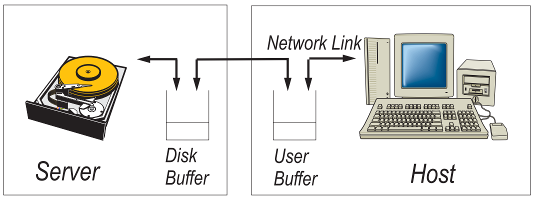


Figure 1. Digital Video Server Storage and Transfer of Video Frames

In our previous paper [1] we have analyzed three models, namely the *Gamma Service Cycle Mean* which captures the variation of data size between delivery cycles (considering each video cycle a stream of constant-size video frames),similar to analyzing video traffic at a Group of Pictures (GoP) level, the *Strand Gamma Frame Size Distribution* model which uses information regarding the frame size probability distribution for the entire video segment and the *Gamma Fractal Noise* model, which gives a better fitting to the actual data. This model captures the self-similar nature of video information not only on its statistical moments but also on its probability distribution. In this paper we provide additional details regarding the Gamma Fractal Noise model. We develop easy to compute asymptotics based on the Chernoff bound for the buffer size and the service rate. We compare the performance of this stochastic model with simulations of real traces generated by the MPEG-4 video encoder.

In Section 2, we describe the video server storage scheme and the video encoder, whereas in Section 3 we discuss the main operational parameters. In Section 4, we establish the typical mean service rate parameters

[1] Computer Science Department. rramirez@itesm.mx
[2] Centro de Electrónica y Telecomunicaciones. rmrodrig@itesm.mx

used on the analysis of video servers. The performance analysis is done in Section 5 with the Chernoff bound and the calculations for the three-parameter Gamma distribution, along with section 6, where we analyze the Gamma Fractal Noise Model. The conclusion and some of the proofs are in Section 7 and Appendices, respectively.

2. DIGITAL VIDEO SERVER SCHEME

2.1. Storage of Multiple Videos

In this DVSS the video files are stored in a single disk, and when a user requests one, the different frames that make up the movie are retrieved one-by-one in a sequential order. Once a video frame has been retrieved from the disk, it will be delivered through a network link to the user according to the playback rate stated in frames per second.

Basically, our storage layout is the same as the one studied in [2]. This means that each block of $n_{v,i}$ display units (typically frames) is separated from the next and the previous block by exactly $I_{v,i}$ seconds for video strand i. The amount of display units $n_{v,i}$ is called *granularity* and the separation $I_{v,i}$ is called *interleaving.* The separation between blocks leaves a gap that does not necessarily lay empty as it is used to store other videos. The process of storing video files in every gap is called *merging.* Other important parameters are: a) $\overline{S}_{vf,i}$, which is the mean size of the video frame in bytes for video strand i, b) R_{dr}, the hard disk data transfer rate in bits/sec and c) $R_{pl,i}$, the replay rate required by video i in frames per second.

The scattering and the interleaving parameters define a storage pattern in bits of length $L_i = n_{v,i}\overline{S}_{vf,i} + I_{v,i}R_{dr}$, characterized by the pair (M_i, G_i) where $M_i = n_{v,i}\overline{S}_{vf,i}$, and $G_i = I_{v,i}R_{dr}$, (see Fig. 2).

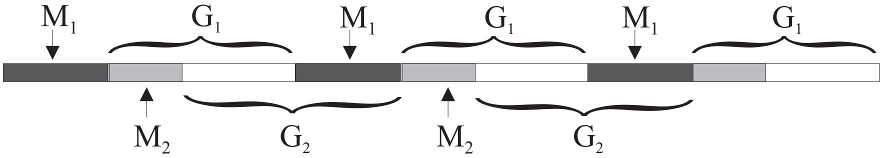


Figure 2. Disk storage pattern in bits.

In a variable bit-rate system, the pairs (M_i, G_i) are of different length due to the fact that an M block is formed by n_{vs} video frames of different sizes. Thus a video stream will consist of the sequence of frames $\{S_{vf,i,k}\}$, which are stored on disk using the merge sequence $\{(M_{i,j}, G_i)\}$, where $M_{i,j} = \sum_{k=j*n_{vs,i}}^{j*n_{vs,i}+n_{vs,i}-1} S_{vf,i,k}$.

2.2. SAS scheme

The video server assigns a separate buffer to each user i, which is attended during a service time l_i (filled with video data during this time). This service time depends on the buffer depletion rate $\mu_{d,i}$. The mean depletion rate is given by $\mu_{d,i} = R_{pl,i}\overline{S}_{vf,i}$, where $\overline{R}_{pl,i}$ is the mean replay rate for user i. This depletion rate represents the video information that is sent from a user buffer in the video server to each video display customer through the network. We assume that the information is sent to the customer at exactly the frame display rate $(R_{pl,i}S_{vf,i})$. In the Switched Arrival System (SAS) scheme, each user buffer receives frames during a period of time called service time l_i. The server then switches to another buffer. It will not service this buffer again until it attends all the other users' buffers once. This is called a service cycle. The service time each buffer is allocated must provide with enough video frames so that the buffer will comply with user play-back rate requirements without frame starvation during the entire service cycle [3], see also Fig. 3.

In the SAS scheme with N users, each buffer i is associated with a service request of that particular user. Let us assume that each buffer i has a capacity b_i, and $x_i(t)$ is the state of the buffer at time t. Let us also consider that $L_{seek,i}$ and $L_{rot,i}$ are the seek delay and plate rotation delay for request i. So, $\alpha_i = L_{seek,i} + L_{rot,i}$ is the switching time for user i. This is the time needed for the disk head to be positioned in the correct cylinder

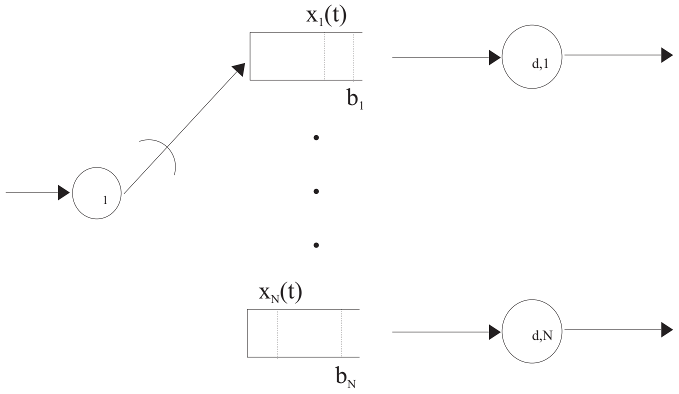


Figure 3. Switch Arrival Service (SAS) scheme of buffers.

and allocation block. Similarly, $\mu_{d,i}(t)$ is the depletion rate as a function of time, whereas $\lambda_i(t) = R_i - \mu_{d,i}(t)$ is the fill rate for buffer i in bits per second, where $R_i = R_{dr}\, f_i$ is the disk throughput for user i, and f_i is a factor which depends on the switching delay between requests, interleaving and number of frames delivered. On the other hand, $l_i(r)$ is the service duration in a cycle r for user i. When the Service Scheduling Policy is periodic, we always have that $l_i(r) = l_i(r+1)$. Finally, we will call a period in which all pending service requests are attended exactly one at a time a *Service Cycle*.

The disk subsystem must attend each request by filling the designated buffer with video frames either up to a certain level or for a fixed period of time. The order and manner in which these requests are attended is called a service scheduling policy. In general, a service scheduling policy $\Phi(r, \mathbf{x})$ is a function which returns a permutation of the order in which the buffers will be serviced depending on cycle r and the state $\mathbf{x} = \mathbf{x}(t)$ of all the buffers. The policy is called *open loop* if the permutation yielded by it depends on time or service cycle k, and it is called *closed loop* when the permutation depends on the state $\mathbf{x}$ of the buffers. If $\phi(t)$ is a buffer which will be serviced at time t, then the service scheduling policy is called *periodic* when $\phi(t) = \phi(t + S)$, where S is a constant service cycle time.

In this paper, we will consider the *Periodic Open Loop* (POL) [5,4] scheduling policy which serves each user i during a constant predefined time l_i, as opposed to the *Periodic Closed Loop* (PCL) scheduling policy which serves each client until $x_i(t) = b_i$.

In this SAS scheme the time delay that a particular data packet experiences in the buffer is not relevant, since as long as there are frames on the buffer ready to be sent, there will be a steady supply of frames exiting the buffer according to the play-back rate. The main focus, however, is the empty state of the buffer. This means we are mainly concerned with characterizing the state in which the disk subsystem did not provide sufficient video frames for all buffers, that is, that the probability that one or some of the buffers will reach the empty state.

2.3. Video Encoder

The MPEG video encoder takes as its input stream a sequence of digitalized frames, each of them containing a two-dimensional array of pixels (pels). In order to specify the data rate of the video encoder, it is important to know the number of frames per second, the number of lines per frame, and the number of pels per line. For each pel, both luminance and chrominance information is stored. The video strands used in this paper are encoded by using MPEG-4 [6], and they are stored in a disk by using a merging pattern. Similarly to MPEG-2, MPEG-4 is a layered encoding scheme. This means that the video data stream consists of a basic layer stream that contains the most important video data and one or more enhancement layers, which can be used to improve the quality of the video sequence. Unlike MPEG-1 and MPEG-2, which are frame based, MPEG-4 is object based. Each scene is composed of Video Objects (VOs) which are individually encoded. The scalability layers of each VO are called Video Object Layers (VOLs). Each VOL consists of an ordered sequence of frames called Video Object

Planes (VOPs). For each VOP, the encoder makes the processing of shape, motion, and texture characteristics. Shape information is encoded by bounding the VO with a rectangular box and then dividing the bounding box into Macro Blocks (MBs). Each MB is classified as lying inside the object, on the objects border, or outside the object but inside the bounding box.

Transforms and entropy coding reduce spatial redundancy, and the prediction of future frames based on motion vectors reduce the temporal redundancy. Spatial and temporal redundancies are reduced by using three types of VOPs:

I (Intracoded) VOPs. The absolute texture values (chrominance and luminance) in each MB are compressed (spatial compression) without references to other VOPs using an approach similar to the one in JPEG encoded images using Discrete Cosine Transform (DCT) and then encoded by lossless variable-length encoders, such as entropy coding or Huffman coding. They allow for random access to video sequences.

P (Predicted) VOPs. Each MB is predicted from the closest match in the preceding I (or P) VOP using motion vectors. The compression ratios of P-VOPs are typically higher than I-VOPs. P-VOPs can be used in subsequent predictions of MBs.

B (Bidirectional) VOPs. Each MB is interpolated from the preceding I (or P) VOP and the succeeding P (or I) VOP. Those VOPs are not used in subsequent predictions and provide the highest compression.

In Table 1 we list the video trace sequences used in this study. Each video file was recorded at a frame rate of 25 frames/sec, with a luminance resolution of 176 × 144 pels and 4:1:1 chrominance subsampling at a color depth of 8 bits. The encoding was done without rate control, so they are Variable Bit Rate (VBR) traces. The entire video stream is one VO and there is only one layer. The Group of Pictures (GOP) pattern was set to *IBBPBBPBBPBB* and the quantization parameters were fixed at 4 for I, B and P frames. There are roughly 90,000 frames per trace (see [7] for a more detailed description of the video traces).

Jurassic Park	Silence of the Lambs	Star Wars IV
Mr. Bean	Star Trek: First Contact	From Dusk Till Dawn
The Firm	Die Hard III	Starship Troopers
Formula 1 car race	Alpine Ski	Soccer European Championship 1996

Table 1. Encoded video sequences

The SAS system feeds these video traces to the different buffers in a round robin manner, meaning that the data corresponding to a particular video segment is transferred to a particular buffer until user-service time is complete or the buffer is full. Then, the SAS system proceeds to the next user. No consideration is made for fairness in video frame delivery.

3. THE GLOBAL TIME BALANCE RELATIONS

The main parameters for the performance of the video server are: The service-rate constant k_i, the maximum number of simultaneous subscribers N_{max} and the user buffer size b_i.

Since all buffers are depleted in parallel, the reading time of data blocks needed for the disk storage subsystem to replenish all buffers must be less than or equal to the minimum buffer emptying time. One buffer will empty faster than the others because either its size is smaller or its playback rate is faster. Assume that buffer i is filled with exactly $k_i n_{v,i}$ frames of size $S_{vf,i,j}$, where $j = 1, 2, ..., k_i n_{v,i}$. We may also assume that buffer i is filled with k_i blocks of video and that each block contains $n_{v,i}$ frames. To comply with the playback-rate demands, relation (1) must hold. We shall call the following inequality (1) the *Global Time Balance Relation*

$$\alpha(N) + \upsilon(N, \mathbf{k}) + \tau(N, \mathbf{k}) \leq \min\left(k_i \frac{n_{v,i}}{R_{pl,i}}\right) \tag{1}$$

where N is the number of concurrent users in a given time, $\mathbf{k} = [k_1, k_2, \ldots, k_N]$, $\alpha_T = \alpha(N) = \sum_{i=1}^{N} \alpha_i$ is the service cycle switching delay, $\upsilon(N, \mathbf{k}) = \sum_{i=1}^{N} \upsilon_i(k_i)$ is the service cycle interleaving time (the time the disk head spends over interleaving gaps) and $\tau(N, \mathbf{k}) = \sum_{i=1}^{N} \tau_i(k_i)$ is the time spent by the disk fetching frames for all subscribers in a service cycle. In our system, this translates to (see [8])

$$N(L_{seek} + L_{rot}) + \sum_{i=1}^{N} \left(k_i I_{v,i} + \frac{\sum_{j=1}^{k_i n_{v,i}} S_{vf,i,j}}{R_{dr}} \right) \leq \min \left(k_i \frac{n_{v,i}}{R_{pl,i}} \right). \tag{2}$$

The total time equals the addition of video frames fetching time, plus the time spent by the disk head over the interleaving space, plus the switching time. The disk throughput for user i can be written as

$$R_i = R_{dr} f_i = R_{dr} \left[\frac{\tau_i(k_i)}{\alpha_i + \upsilon_i(k_i) + \tau_i(k_i)} \right] \tag{3}$$

as we mention above, factor f_i is the proportion of time that the disk spends fetching video frames for user i divided by the total time it spends servicing request i.

We also define the service cycle disk throughput as [3]

$$R_{sc} = R_{dr} f_{sc} = R_{dr} \left[\frac{\tau(N, \mathbf{k})}{\alpha(N) + \upsilon(N, \mathbf{k}) + \tau(N, \mathbf{k})} \right] \tag{4}$$

where f_{sc} is the proportion of time within a complete cycle that the disk needs to transfer video frames.

Notice though, that R_{sc} is not equal to $\sum_{i=1}^{N} R_i$, where R_i is the disk throughput when user i is being serviced. Actually, we would expect that $R_i \approx R_{sc}$, $i = 1, 2, \ldots, N$.

In Fig. 4 we establish the relationship among disk transfer rate, user disk throughput, service cycle disk throughput and buffer state. In Fig. 4a) we illustrate the fact that the disk transfers information only when the disk head is over a granularity section for user i (M_i). On the other hand, the disk subsystem will not transfer any information when the disk head is over a interleaving disk section for user i (G_i). Even though this section contains information for another user, such information will not be read. Also, the disk will not transfer information whenever it is performing seek or rotation operations, (α_i). Thus, the sum of all the times the disk head is over user i in M sectors is called τ_i. We should notice that τ_i is a function of k_i since it depends on the number of (M_i,G_i) pairs that will be in the disk head path to finish a user i turn. Consequently, the sum of all the times the disk head is over G sectors is called υ_i which is also a function of k_i. The sum of switching, interleaving and granularity times for user i is called $\zeta_i = \zeta_i(k_i) = \alpha_i + \tau_i + \upsilon_i$. This constitutes a user i *service cycle time*, see Fig. 4b. We should remember that R_{dr} is the maximum disk data transfer rate. However, for each disk head switching time from user to user (seek and rotation operations) and each time that the disk head is over interleaving sections we waste disk bandwidth and therefore the disk throughput $R_i < R_{dr}$.

The dotted-lines in Figs. 4b) and 4c) show the disk throughput R_i. To compute this throughput for an individual user turn, we add the amount of information in bytes that has been transferred to user-video buffer and then divide by the duration of the turn, ζ_i (Fig. 4b). To compute the service cycle disk throughput R_{sc}, the information transferred to all users is added and then divided by the duration S of the service cycle (Fig. 4c).

The behavior of the buffers $x_i(t)$ is shown in Fig. 4d). We should observe that when the disk subsystem is attending a user its buffer is not filled at rate R_i, rather it is filled at rate $\lambda_i = R_i - \mu_{d,i}$. This is so because the buffer is emptied at rate $\mu_{d,i}$ bytes per second at the same time. At any given moment only one user buffer has been filled, and all other buffers are depleted.

We can define the *system load* ρ as the ratio between the accumulated video data demand and the disk throughput (R). Namely,

[3] On the rest of this paper, R_{sc} will be referred just as R.

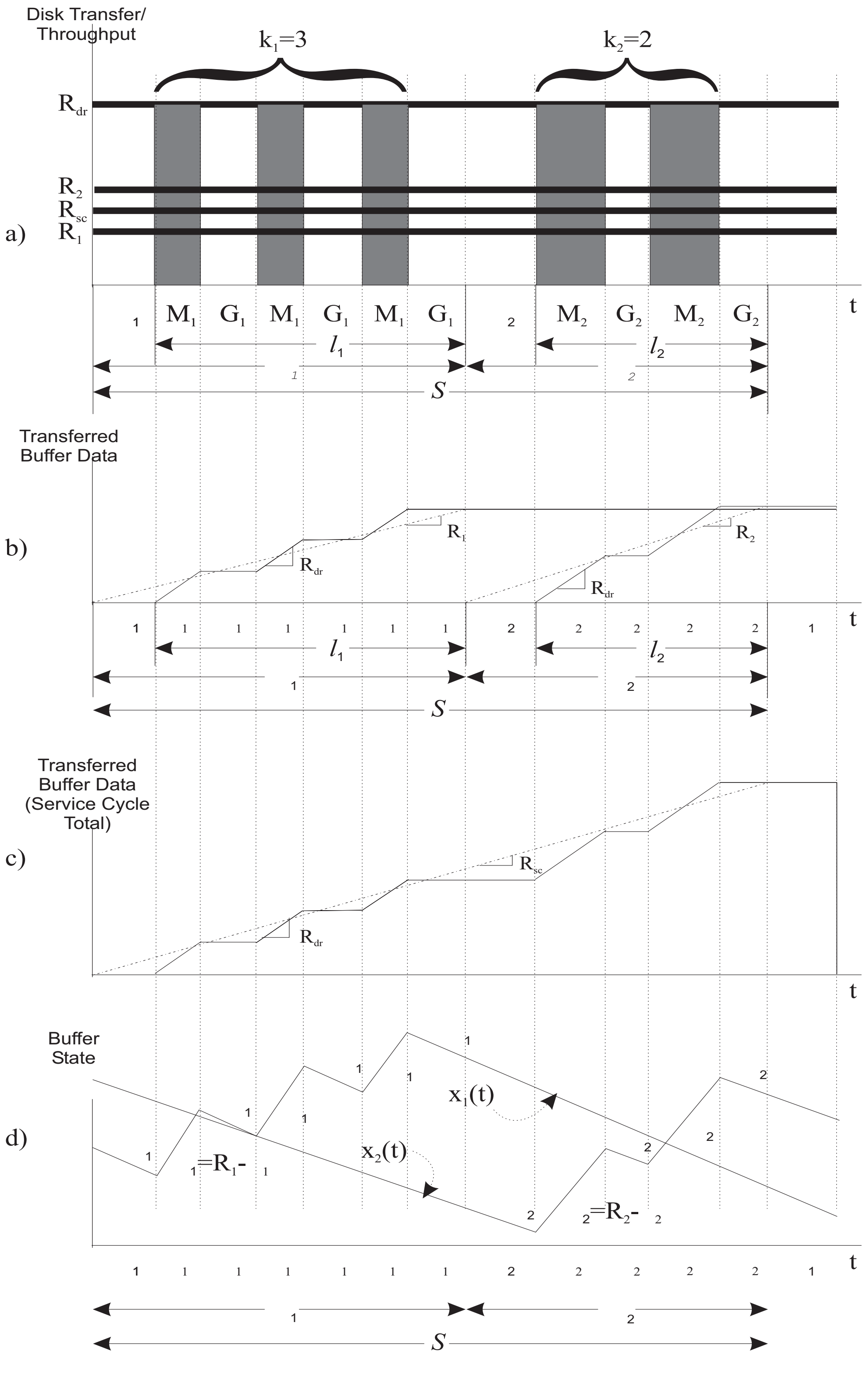


Figure 4. Relation among disk transfer rate, user and service cycle disk throughput and buffer state for N=2 users

$$\rho = \frac{1}{R}\sum_{i=1}^{N} \mu_{d,i}. \tag{5}$$

Another important parameter is N_{max}, which is the maximum number of subscribers that a scheduling policy Φ can give service to simultaneously.

These operational parameters can be related as it is stated on the three propositions explained in the rest of this Section.

The global time balance relation is in fact equivalent to the system load. Let us invert inequality (1),

$$\frac{1}{\alpha(N) + \upsilon(N,\mathbf{k}) + \tau(N,\mathbf{k})} > \max\left(\frac{R_{pl,i}}{k_i n_{v,i}}\right) i = 1, 2, \ldots, N \tag{6}$$

Now, multiplying inequality (6) by $\tau_i(k_i)$, we obtain

$$\frac{\tau_i(k_i)}{\zeta(N) + \upsilon(N,\mathbf{k}) + \tau(N,\mathbf{k})} > \max\left(\frac{\tau_i(k_i) R_{pl,i}}{k_i n_{v,i}}\right) = \max\left(\frac{R_{pl,i}\overline{S}_{vf,i}}{R_{dr}}\right) \tag{7}$$

since $\tau_i(k_i) = \sum_{j=1}^{k_i n_{v,i}} S_{vf,i,j}/R_{dr}$ it can be closely approximated by $k_i n_{v,i}\overline{S}_{vf,i}/R_{dr}$. Taking summation from 1 to N on both sides of inequality (7), we find

$$R_{dr}\sum_{i=1}^{N}\frac{\tau_i(k_i)}{\alpha(N) + \upsilon(N,\mathbf{k}) + \tau(N,\mathbf{k})} = R_{dr}\frac{\sum\limits_{i=1}^{N}\tau_i(k_i)}{\alpha(N) + \upsilon(N,\mathbf{k}) + \tau(N,\mathbf{k})} > \sum_{i=1}^{N}\mu_{d,i} \tag{8}$$

since $\mu_{d,i} = \sum_{i=1}^{N} R_{pl,i} S_{vf,i}$. The ratio on the left hand side of the inequality is f_{sc} and since $R = R_{dr} f_{sc}$, we show the equivalence. We now can establish the following proposition

Proposition I. There exist values $k_i = 1, 2, \ldots, N$ such that inequality (1) holds if and only if $\rho < 1$.

In a switching system, if the disk data rate cannot be increased, the net disk throughput can be increased either by reducing the switching delay or by increasing the user service time. However, it is not possible to reduce the switching time in a SAS system because of the physical limitations of disk seek and of rotation operations. Therefore, it is only possible to increase the user service time up to a certain level in order to keep the fairness level for all users. So we should ask: How much should we increase user service time in order to attend the largest number of concurrent users?. For instance, decreasing the switching delay to zero is equivalent to having only one user in the system. It is a somewhat surprising result that we can use the service rate achieved by having only one user on the system to compute the maximum number of concurrent subscribers that can be serviced, as will be shown below. We must remember that service time for user i is represented by k_i.

When a new user enters the system, the server must increase the service rate to adequately attend all current users. Since we consider the disk transfer rate R_{dr} as a constant, then the disk can only do this by reducing switching time between users, implying a higher k_i for each user. As already mentioned in [4], increasing the switching time decreases the user service-rate, whereas decreasing the switching time increases the user service-rate. This can be appreciated if we look at Eq. (3) in the following manner

$$R_i \approx R_{dr}\left[\frac{\frac{k_i n_{v,i}\overline{S}_{vf,i}}{R_{dr}}}{\alpha_i + k_i I_{v,i} + \frac{k_i n_{v,i}\overline{S}_{vf,i}}{R_{dr}}}\right] = R_{dr}\left[\frac{\frac{n_{v,i}\overline{S}_{vf,i}}{R_{dr}}}{\frac{\alpha_i}{k_i} + I_{v,i} + \frac{n_{v,i}\overline{S}_{vf,i}}{R_{dr}}}\right] = R_{dr}\left[\frac{M_i}{R_{dr}\frac{\alpha_i}{k_i} + (G_i + M_i)}\right] \tag{9}$$

In Eq. (9) we can observe that as the system increases k_i for each user i, then the disk throughput asymptotically approaches $R_{dr}[M_i/(G_i + M_i)]$ which we define as R°, i.e. the maximum data transfer rate allowed by the merged storage scheme, which is achieved when there is a single user in the system. As the number of users increases the values of R_i increase as well. However, they will never be equal to R° no matter how large k_i can be.

Since having $k_i = \infty, i = 1, 2, \ldots, N$ is equivalent to having no switching delay. Then from Fig. 4 we have

$$\lim_{\mathbf{k}\to\infty} \rho(N, \mathbf{k}) = \lim_{\mathbf{k}\to\infty} \frac{\sum\limits_{i=1}^{N} \mu_{d,i}}{R_{dr}\left[\frac{\sum_{i=1}^{N} k_i M_i}{\sum_{i=1}^{N} R_{dr}\alpha_t + k_i(G_i + M_i)}\right]} = \frac{\sum\limits_{i=1}^{N^\circ} \mu_{d,i}}{R_{dr}\left[\frac{\sum_{i=1}^{N^\circ} M_i}{\sum_{i=1}^{N^\circ} (M_i + G_i)}\right]} = \frac{\mu_d}{R^\circ} = \rho^\circ \tag{10}$$

Now, by setting $\rho^\circ = 1$, we can find N_{max}. If N° is an integer number, then $N_{max} = N^\circ - 1$, and if N° is a real number, then $N_{max} = \lfloor N^\circ \rfloor$, which allows us to establish the following proposition:

Proposition II. N_{max} is found by computing N when the service-rate parameters k_i, become infinite.

We can compute the buffer capacity of each video stream when the scheduling policy is periodic. For our purposes, we have used the Round Robin periodic policy. By computing the data coming in and out of the buffers, the buffer level x_i at time t_{m+1} can be found with the following recursive equation [4, Section 5.1]:

$$x_i(t_{m+1}) = \lceil x_i(t_m) + \zeta_i(R_i - \mu_{d,i}) - \mu_{d,i} \sum\nolimits_{j \neq i}^{N} (\alpha_j + l_j) \rceil^+ \tag{11}$$

where $\lceil u \rceil^+ = \max\{u, 0\}$.

Assume that the sequence $x_i(t_m)$ converges to a limit point called x_i. Let ζ_i^* be the unique user service cycle in which such limit is obtained. In order to avoid buffer starvation or overflow, the data coming in and out of user buffers must be balanced. Thus from Eq. (11) we find that the necessary and sufficient condition for a non-starving and non-overflowing cycle that can attain such a limit point is (again [4,Section 5.1])

$$\zeta_i R_i = \mu_{d,i} S, \tag{12}$$

which can only mean that

$$\Delta x_i(t_{m+1}) = x_i(t_{m+1}) - x_i(t_m) = 0. \tag{13}$$

Since the limit point in each service cycle remains the same, we need the buffer to be able to hold the video data delivered in the convergence service cycle, that is $b_{min,i} = \zeta_i^* \lambda_i$. Furthermore, let us call φ_i the size of video frames fetched from disk for user i at such service cycle. Then, $\zeta_i^* = \varphi_i / R_i$. Since $\lambda_i = R_i - \mu_{d,i}$, by having $R_i \gg \mu_{d,i}$, we get $b_{min,i} \approx \varphi_i$. Thus the following proposition can be stated.

Proposition III. For a periodic open loop scheduling policy, there exist buffer capacity values called $b_{min,i} = \zeta_i^* \lambda_i$, which are the steady long-term values of the buffer level $\mathbf{X}(t_m) = [x_1(t_m), x_2(t_m), \ldots, x_N(t_m)]$ at the beginning of cycle t_m.

4. MEAN SERVICE RATE PARAMETERS

In the *Quality Proportional (QP)* model the number of blocks transferred to a customer is proportional to the playback rate $k_i = kR_{pl,i}$ [8]. We call parameter k the proportional constant or more specifically the *Proportional Service-Rate Constant.* This parameter specifies the disk throughput for each video strand, meaning that in the average $k\overline{R}_{pl}\overline{n}_v\overline{S}_{vf}$ bytes will be transferred to the user buffer in each service cycle.

We call R the mean disk thoughput (as a function of k)

$$R = R_{dr}\left[\frac{\frac{k\overline{R}_{pl}\overline{n}_v\overline{S}_{vf}}{R_{dr}}}{k\overline{R}_{pl}\left(\overline{I}_v + \frac{\overline{n}_v\overline{S}_{vf}}{R_{dr}}\right) + \overline{\alpha}}\right]. \tag{14}$$

By balancing out the average disk throughput R with the average depletion rate of all buffers $\mu_d = N\overline{R}_{pl}\overline{S}_{vf}$ and taking into consideration the stability condition $\rho = \mu_d/R < 1$ (see Proposition I), we can find $\overline{k}$, as follows (which is basically the same found in [8]),

$$\overline{k} \geq \left\lceil \frac{N\overline{\alpha}}{\overline{n}_v - N\overline{R}_{pl}\left(\overline{I}_v + \frac{\overline{n}_v\overline{S}_{vf}}{R_{dr}}\right)} \right\rceil. \tag{15}$$

Using Proposition II we can find a mean value for the maximum number of concurrent subscribers as follows (which was also found using another method in [8])

$$\overline{N}_{\max} = \left\lfloor \frac{R_{dr}\overline{n}_v}{\overline{R}_{pl}\left(\overline{I}_v R_{dr} + \overline{n}_v\overline{S}_{vf}\right)} \right\rfloor. \tag{16}$$

And using Proposition III, the buffer size for video strand i is given by (which is equivalent to the expressions found in [8] and [4])

$$\overline{b}_i \geq \overline{k}\,\overline{R}_{pl,i}\overline{n}_{v,i}\overline{S}_{vf,i}. \tag{17}$$

5. CHERNOFF BOUND

The Chernoff bound is useful in finding an approximate tail probability bound. This computation is expressed in terms of an exponential probability decay function, which is called the rate function in the large deviations theory. We need to know

$$P(X_1 + \ldots + X_n \geq \varphi = na) \leq e^{-n\ell(a)} \tag{18}$$

for $a > \mathbf{E}[X_1]$, where

$$\ell(a) = -\ln\left(\inf_{\theta} e^{-\theta a}M(\theta)\right) = \sup_{\theta}\left(\theta a - \ln M(\theta)\right), \tag{19}$$

and $M(\theta) = \mathbf{E}\left[e^{\theta X_1}\right] < \infty$, for θ in some neighborhood of 0.

We define $\phi = P(X_1 + ... + X_n \geq \varphi = na)$. For instance, ϕ can equal the probability that the time required to deliver n frames $(X_1, \ldots, X_n)$ from disk will be equal to or greater than the allocated time φ. We can also formulate the problem as follows: "Find a suitable disk fetch time φ such that the disk subsystem will be saturated, and hence it will fail to deliver the required information with probability ϕ (very small) at most".

Now suppose that the video files are stored in such a way that the units of information (video frames) to be retrieved from the disk have a Gaussian *pdf* (in accordance with the Central Limit Theorem) with mean μ_N and variance σ_N^2. The rate function for such a random variable is

$$\ell(a_N) = \frac{1}{2}\left(\frac{a_N - \mu_N}{\sigma_N}\right)^2. \tag{20}$$

Hence,

$$a_N = \mu_N + \sigma_N\sqrt{\frac{-2\ln(\phi)}{n}}. \tag{21}$$

a_N can be thought of as the individual contribution of each random variable to the accumulated service time.

On the other hand, if we assume a two parameter Gamma *pdf* for the random variable X to model the units of information (as suggested in [9, 10, 11 and 12]), with scale parameter λ and shape parameter α, where by matching moments we find the estimators $\mathbf{E}[X] = \alpha/\lambda$ and $\mathbf{Var}[X] = \alpha/\lambda^2$.

$$\ell(a_g) = a_g\lambda - \alpha - \alpha \ln\left\{\frac{a_g}{\mathbf{E}[X]}\right\}. \tag{22}$$

where a_g has the same meaning as a_N.

6. STOCHASTIC MODEL FOR DVSS OPERATIONAL PARAMETERS

6.1. Basic Parameters

In the following sections we assume that all stored video files have the same playback rate, granularity and mean frame size but not necessarily the same variance and that the DVSS follows a *QP* service policy. We will also consider the switching delay α as a constant for all users, as well as deterministic interleaving.

In order to be able to define stochastic models for service rate parameters (mainly k), we begin by writing the global time balance relation (1) as

$$\alpha(N) + \upsilon(N,k) + \tau(N,k) \le kn_v^{(\min)};\quad n_v^{(\min)} = \min\{n_{v,i}\}\ \forall i \tag{23}$$

where k is the unique service-rate proportionality constant as established in the *QP* model of Section 4.

The disk throughput is now

$$R = R_{dr}\left[\frac{\tau(N,k)}{\alpha(N) + \upsilon(N,k) + \tau(N,k)}\right]. \tag{24}$$

We also define, in accordance with Proposition II, the maximum number of subscribers as

$$N_{\max} = \lim_{k\to\infty} \frac{R_{dr}}{\overline{\mu}_d}\left[\frac{\tau(N,k)}{\alpha(N) + \upsilon(N,k) + \tau(N,k)}\right], \tag{25}$$

where $\overline{\mu}_d$ is the mean user frame-depletion rate.

The *Success Probability* (P_{succ}) is the probability that all required video frames for all users in a service round will be read and placed on their respective buffers on time. The *Saturation Probability* (P_{sat}) is the probability that the disk throughput will be exceeded by the accumulated demand bit rate. The Saturation Probability is defined as

$$P_{sat} = \phi = P[\mu_d > R] = 1 - F_{\mu_d}(R) = 1 - P_{succ}. \tag{26}$$

In accordance to Proposition I, the Saturation Probability can also be defined as

$$P_{sat} = P\ [\alpha(N) + \upsilon(N,k) + \tau(N,k) > kn_v^{(\min)}\ ]. \tag{27}$$

By Proposition III and the Chernoff bound of the video frames size, $b_i = na = kR_{pl}n_v a$.

We characterize $\tau(N,k)$ in the following manner: let $Y_i = \{Y_i(t), t > 0, i = 1, \ldots, n\}$ be the continuous-time random process where $Y_i(t)$ is the amount of video information inserted in user buffer i at time t by the disk. Let $X_i(t) = Y_i(t) - Y_i(t-1)$ be its strictly positive increment process. We will assume that $Y_i(t)$ has stationary increments. Also $\mathbf{E}[X_i(t)] = \overline{S}_{vf,i}$ and $\mathbf{Var}[X_i(t)] = \sigma_i^2$. The process of filling user buffers conforms to the discrete accumulated time series $X_i^{(m)} = \sum_{t=1}^m X_i(t) = Y_i(m) - Y_i(0)$. Thus $\tau(N,k) = \sum_{i=1}^n X_i^{(m)}/R_{dr}$.

6.2. Mathematical Description of Self-Similarity

There are several, yet not equivalent, definitions of self-similarity. The standard one states that a continuous-time process $Y = \{Y(t), t \geq 0\}$ is self-similar (with self-similarity or Hurst parameter H) if it satisfies condition [13]

$$Y(t) \triangleq a^{-H} Y(at),\ t \geq 0,\ a > 0,\ 0.5 < H < 1 \tag{28}$$

where equality is in the sense of finite-dimensional distributions. It is important to note that with $t = 1$ and $a = t$, Eq. (28) also means that

$$Y(t) \triangleq t^{H} Y(1) \tag{29}$$

and

$$f_{Y(1)}(x) = |t^{H}| f_{Y(t)}(|t^{H}|x), \tag{30}$$

where $f_{Y(t)}(x)$ is the *pdf* of $Y(t)$. Also, it is assumed that

$$\mathbf{E}[Y^{2}(t)] = \sigma^{2} t^{2H} \tag{31}$$

where $\sigma^{2} = E[Y^{2}(1)]$. We should notice that Eq. (31) states for self-similar variable aggregation, that the resulting variance does not scale linearly, leading to interesting behaviors, such as lack of smoothness of the resulting aggregated traffic, which complicates the buffer dimensioning. This is one of the main reasons why it is so difficult to predict DVSS operational parameters. For our discrete-time stochastic process, it translates to

$$\sigma^{(n)} = n^{H} \sigma,\ 0.5 < H < 1. \tag{32}$$

6.3. Gamma Fractal Noise

We assume that for a single video stream, the frames that read on a single service cycle will have different size. This means that for each user, the size of all the video frames read is the sum of $kR_{pl}n_{v}$ random variables. We can see from the histograms in Fig. 5 that most video traces are not symmetric and they can be fitted by a Gamma *pdf* as noted by several authors as well, see [9, 11 and 12].

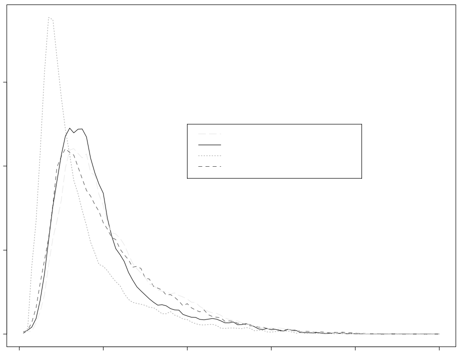


Figure 5. Histogram for the video frame size of four video traces.

It is well known that we obtain a Gamma random variable after adding an arbitrary number of Gamma random variables with the same scale parameter. We assume that video streams for different users have the same frame size mean and different variance, so the correct distribution of the sum is not necessarily a Gamma random variable. However, we can obtain an approximate Gamma random variable after considering the same variance for the video streams of different users. The common variance that we will consider is the largest variance of all the video streams.

The global time balance inequality can now simply be written as

$$N\alpha + NkR_{pl}\overline{I}_v + \frac{\varphi}{R_{dr}} \leq kn_v^{(\min)}, \tag{33}$$

where φ is the sum of the sizes of all the videos that must be read and delivered in order to comply with the saturation probability, and inequality (33) is achieved with probability ϕ.

In this model, which is closely related to the one found in [14], we compute φ using a continuous-time process, say $Y(t)$, and later tie it to the discrete-time increment process $X(n)$. Let $Y(t) = \mu t + Z(t)$, with $\mathbf{E}[Y]$=μt. The stochastic process $Z(t)$ is self-similar with $\mathbf{E}[Z(t)] = 0$ and $\mathbf{E}[Z^2(t)] = \sigma^2 t^{2H}$. The accumulation process can be written as [15]

$$\Sigma_{i=1}^{n} X(i) = Y(n) - Y(n-1) + Y(n-1) \ldots - Y(0) = n\mu + Z(n) \triangleq n\mu + n^H Z(1). \tag{34}$$

The discrete-time random process X(n) is a collection of random variables $\{X(1), X(2), \ldots,\}$ and it will be denoted as $\{X_1, X_2, \ldots,\}$. Thus, the Chernoff limit for the saturation probability is

$$\phi = P(X_1 + \ldots + X_n \geq \varphi = na) \leq e^{-n\ell(a)}. \tag{35}$$

Also, from self-similarity, the video frame accumulation probability is

$$P(X_1 + \ldots + X_n > \varphi) \triangleq P(n\mu + n^H Z(1) > \varphi) = P\left(Z(1) > \frac{\varphi - n\mu}{n^H} = a\right). \tag{36}$$

We will denote $Z(1)$ as the random variable Z. The marginal distribution of $Z(t)$ is characterized by a three parameter Gamma *pdf* with parameters: shape α, scale λt^{-H}, and location $-\alpha/\lambda t^H$. It is important to note that by making $t = 1$ in $Z(t)$ and from the self-similarity property we have $\mathbf{E}[X_i]$=$\mu_G = \alpha/\lambda$ and $\mathbf{Var}[X_i]$=$\sigma_G^2 = \alpha/\lambda^2$. In Appendix A we show that under these conditions, $Z(t)$ is self-similar.

The *moment generating function* of such a three-parameter Gamma random variable is the same as the one of a two-parameter Gamma random variable shifted by A

$$M(\theta) = \mathbf{E}[e^{\theta Z}] = \left(\frac{\lambda}{\lambda - \theta}\right)^{\alpha} e^{\theta A} \tag{37}$$

Then according to the Chernoff bound we have

$$P[Z \geq a] \leq \min_{\theta > 0} e^{-\theta a} \left(\frac{\lambda}{\lambda - \theta}\right)^{\alpha} e^{\theta A} = e^{-\ell(a)}, \tag{38}$$

where

$$\ell(a) = \sup_{\theta \geq 0} \left\{\theta a - \ln M(\theta)\right\}. \tag{39}$$

Since Z is a zero mean random variable, $A = -\mu_G$. We find that $\theta^* = \min_{\theta \geq 0} = \lambda - \frac{\alpha}{a + \mu_G}$ and thus

$$\ell(a) = \lambda(a + \mu_G) - \alpha - \alpha \ln\left\{1 + \frac{a}{\mu_G}\right\}. \tag{40}$$

From Eqs. (35) and (40), after some algebra we find that

$$\ln(\phi) \leq \alpha + \alpha \ln\left\{1 + \frac{a}{\mu_G}\right\} - \lambda(a + \mu_G) \tag{41}$$

which does not yield an implicit solution for a. However, we can find a good approximate solution by considering that "a" in the term $\lambda(a+\mu_G)$ is more important than "a" in the term $\log(1+a/\mu_G)$. Thus we can redefine Eq. (41) as

$$a \leq \mu_G \left(\ln\left\{1 + \frac{a'}{\mu_G}\right\}\right) - \frac{\ln(\phi)}{\lambda} \tag{42}$$

where a' can be approximated with the Gaussian case in Eq. (21). Let us denote by the subscript N the parameters for this Gaussian case, thus

$$a_N = a' \approx \sigma_N \sqrt{-2\ln(\phi)}. \tag{43}$$

We should remember that $\mathbf{E}[Z(t)] = 0$. Now, by substituting a' given in (43) into inequality (42) we obtain

$$a_G \leq \mu_G \left(\ln\left\{1 + \frac{\sigma_N\sqrt{-2\ln(\phi)}}{\mu_N}\right\}\right) - \frac{\ln(\phi)\sigma_G^2}{\mu_G}. \tag{44}$$

Now, by matching the moment estimators for the mean and variance we can assume that $\sigma_G = \sigma_N = \sigma$, $\mu_G = \mu_N = \mu$, and by letting $a_G = a$ we obtain the following approximated explicit solution for a

$$a = \frac{\varphi - n\mu}{n^H} \lesssim \mu \left(\ln\left\{1 + \frac{\sigma\sqrt{-2\ln(\phi)}}{\mu}\right\}\right) - \frac{\ln(\phi)\sigma^2}{\mu}. \tag{45}$$

Also, by taking into account that $n = NkR_{pl}n_v$, and $\mu = S_{vf}$ we find that a reasonable approximation for φ is given by

$$\varphi = na \leq NkR_{pl}n_vS_{vf}\left[1 + (NkR_{pl}n_v)^{H-1}\left(\ln\left\{1 + \frac{\sigma_{\max}\sqrt{-2\ln\phi}}{S_{vf}}\right\}\right)\right] - \frac{\ln(\phi)\,(NkR_{pl}n_v)^H\,\sigma_{\max}^2}{S_{vf}}. \tag{46}$$

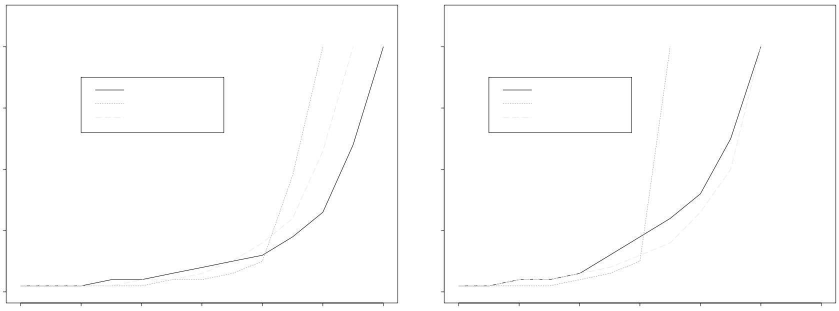
6.4. DVSS Operational Parameters

Substituting Eq. (46) in Eq. (33) we find

$$\begin{gathered} N\alpha + NkR_{pl}\overline{I}_v + \tfrac{NkR_{pl}n_vS_{vf}}{R_{dr}}\left[1 + (NkR_{pl}n_v)^{H-1}\left(\ln\left\{1 + \tfrac{\sigma_{\max}\sqrt{-2\ln\phi}}{S_{vf}}\right\}\right)\right] \\ -\tfrac{\ln(\phi)(NkR_{pl}n_v)^H\sigma_{\max}^2}{S_{vf}R_{dr}} \leq kn_v^{(\min)}. \end{gathered} \tag{47}$$

As it is the case for a en Eq. (41), it is not possible to find an explicit expression for k in Eq. (47). So we must use Eq. (15) for all the instances of k with H as its power (and call them $\overline{k}$), and optionally use the numerical refinement procedure shown below in this Subsection. Then, using simple algebra and Proposition I we find

$$k \geq \left\lceil \frac{N\alpha - \frac{\ln(\phi)\sigma_{\max}^2\left(NR_{pl}\overline{n}_v\overline{k}\right)^H}{R_{dr}\overline{S}_{vf}}}{n_v^{(min)} - NR_{pl}\overline{I}_v - \frac{NR_{pl}\overline{n}_v\overline{S}_{vf}}{R_{dr}}\left[1 + \left(NR_{pl}\overline{n}_v\overline{k}\right)^{H-1}\ln\left\{1 + \frac{\sigma_{\max}}{\overline{S}_{vf}}\sqrt{-2\ln(\phi)}\right\}\right]} \right\rceil. \tag{48}$$



(a) Saturation probability = 1×10^{-2}

(b) Saturation probability = 1×10^{-6}

Figure 6. Proportionality constant k as a function of the number of active subscribers. This figure compares the GFN model and the earlier GCM model versus simulation data results.

Now, according to Proposition III, the buffer size for each video strand is simply φ/N. We also use each individual σ_i to gain further accuracy on individual buffer capacity determination

$$b_i = kR_{pl}\overline{n}_v\overline{S}_{vf}\left(1 + \left(NkR_{pl}\overline{n}_v\right)^{H-1}\ln\left\{1 + \frac{\sigma_i}{\overline{S}_{vf}}\sqrt{-2\ln(\phi)}\right\}\right) - \frac{\ln(\phi)\sigma_i^2 N^{H-1}\left(kR_{pl}\overline{n}_v\right)^H}{\overline{S}_{vf}}. \tag{49}$$

As shown in Appendix B, the FGN model is not only self-similar, but also long-range dependant.

In Fig. 6 we show several plots of k versus N for the models GFN and our previous GCM [1] in comparison with the simulation data results for $P_{sat} = 1 \times 10^{-2}$ and $P_{sat} = 1 \times 10^{-6}$. In both figures we clearly see that the GFN model renders an improvement on the prediction for the value of k around the middle ranges of simultaneous subscribers. But more importantly, in Fig. 6b we are able to see that the GFN model follows the simulation data curve in most of the cases, although it still underestimates k within the middle range.

Equation (48) provides a very good approximation for the service-rate proportionality constant k, and it can be made even better by using the simple numerical refinement algorithm that follows:

1.
$$k^{(0)} = \frac{N\overline{\alpha}}{\overline{n}_v - N\overline{R}_{pl}\left(\overline{I}_v + \frac{\overline{n}_v\overline{S}_{vf}}{R_{dr}}\right)}$$

2.
$$i = 0$$

3.
$$k^{(i+1)} = \frac{N\alpha - \frac{\ln(\phi)\sigma_{\max}^2\left(NR_{pl}\overline{n}_v k^{(i)}\right)^H}{R_{dr}\overline{S}_{vf}}}{n_v^{(min)} - NR_{pl}\overline{I}_v - \frac{NR_{pl}\overline{n}_v\overline{S}_{vf}}{R_{dr}}\left[1 + \left(NR_{pl}\overline{n}_v k^{(0)}\right)^{H-1}\ln\left\{1 + \frac{\sigma_{\max}}{\overline{S}_{vf}}\sqrt{-2\ln(\phi)}\right\}\right]}$$

4.
$$i = i + 1$$

5. Repeat 3 and 4 until convergence

6.

$$k = \lfloor k^{(last)} \rfloor$$

As $k^{(0)}$ is already a good approximation, the algorithm converges in just a few steps.

7. CONCLUSIONS

As we have shown, the gamma probability distribution is a good descriptor for the behavior of video data. It enables the creation of simple models that take into account the short-range dependencies inherent in a non heavy-tailed probability distribution, but also the long-range dependencies that often arise in self-similar traffic. It also has the advantage of easy estimation of statistical parameters.

Since the models we have been using rely on the Chernoff bound, which tries to approximate the tail of a probability distribution, it is not surprising that we should obtain better results at low saturation probability requirements and higher user loads. Also, we can see that incorporating the self-similarity characteristics attributed to video data along with the usage of a skewed probability distribution such as Gamma, renders a good improvement on the prediction of digital video server behavior over models that do not take into account these characteristics.

Self-similarity in video data implies a long-range dependence characteristic that has important implications on the development of video data models, such as absence of the Markov memoryless property and the inability to expect smooth behavior under periodic averaging. The simplicity of the model we developed enables easy computation of important operational parameters, such as user service-rate (though the proportionality constant) and buffer size. This computation can be easily extended for admission control, network link bandwidth estimation, network buffer-delay and other.

Also, we have shown that the correct estimation of user buffer allows an important simplification of scheduling policies. For example, we have used Round-Robin, which by nature ensures fairness, and by reducing scheduling over-head, ensures high performance.

The considerations above, when taken into account for the development of not just video, but disctinct network data delivery models, will enable much a better design of the data services that must be understood and produced in the near future.

REFERENCES

1. Raúl V. Ramírez-Velarde and Ramón M. Rodríguez-Dagnino. “Performance Analysis of a VBR video server with Gamma distributed MPEG data”. SPIE, Performance and Control of Next Generation Communication Networks, vol. 5244-16, Orlando, Florida, USA, pp. 131-142, September 2003.
2. P. V. Rangan and H. M. Vin. “Designing File Systems for Digital Video and Audio”. 13^{th} Symposium on Operating Systems Principles, vol. 25, No. 5, pp. 69-79. October 1991.
3. C. Chase and P. J. Ramadge. “Periodicity and Chaos from Switched Flow Systems: Contrasting Examples of Discretely Controlled Continous Systems”, in IEEE Trans. on Automatic Control, Vol. 38, No. 1, pp. 70-83. January 1993.
4. J. C. Yee and P. Varaiya. “Models and Performance of Real-Time Disk Access Policies” in Computer Communications, 18(10):725-741, Oct 1995.
5. J. C. Yee and P. Varaiya. “An Analytical Model for real-Time Multimedia Disk Scheduling”. 3^{rd} International Workshop on Network and Operating Systems Support for Digital Audio and Video, Springer-Verlang, pp. 276-288. November 1992.
6. R. Koenen (Editor). “Overview of the MPEG-4 Standard”. ISO/IEC JTC1/SC29/WG11 N4496, March 2002.
7. H. P. Fitzek and M. Reisslein. “MPEG-4 and H.263 Video Traces for Network Performance Evaluation”, Technical University Berlin, Technical Report TKN-00-06, October 2000.

8. H. M. Vin and P. V. Rangan. “Designing a Multiuser HDTV Storage Server”. IEEE Journal on Selected Areas in Communications, Vol. 11, No. 1, pp. 153-164. January 1993.
9. A. Chodorek and R. D. Chodorek. “Characterization of MPEG-2 Video Traffic Generated by DVD Applications”. 1^{st} European Conference on Universal Multi-Service Networks, pp. 62-70. ECOMN 2000.
10. M. Garret and W. Willinger. “Analysis, Modeling and Generation of Self-Similar VBR Video Traffic”. ACM SigComm, London, pp. 269-280. September 1994.
11. O. Rose. “Statistical Properties of MPEG Video Traffic and their Impact on Traffic Modeling in ATM Systems”. IEEE Conference on Local Computer Networks, pp. 397-406. October 1995.
12. U. K. Sarkar, S. Ramakrishnan and D. Sarkar. “Segmenting Full-Length VBR Video into Shots for Modeling with markov-Modulated Gamma-Based Framework”. Internet Multimedia Management Systems II, SPIE vol. 4519, pp. 191-202. 2001.
13. J. Beran, R. Sherman, M. S. Taqqu and W. Willinger. “Long-range Dependence in Variable-Bit-Rate Video Traffic”. IEEE Trans. on Communications, vol. 43, No. 2-4, pp. 1566-1579. April 1995.
14. I. Norros. “A Storage Model with Self-Similar Input”. Queueing Systems: Theory and Applications, vol. 16, pp. 387-396. 1994.
15. K. Park and W. Willinger. “Self-Similar Traffic and Performance Evaluation”. John Wiley and Sons, Inc. New York, USA, 2000.
16. A. Leon-Garcia. “Probability and Random Processes for Electrical Engineering”. Addison-Wesley 1994.
17. M. S. Taqqu. “A representation for self-similar processes”. Stochastic Processes and their Applications, vol. 7, pp. 55-64. 1978.

8. APPENDIX A. SELF-SIMILARITY USING A THREE-PARAMETER GAMMA PROBABILITY DENSITY FUNCTION

A random variable Z characterized by a three-parameter Gamma *pdf* has the following form

$$f_Z(z) = \frac{B}{\Gamma(C)} \left(B\left(z - A\right)\right)^{(C-1)} e^{-B(z-A)} \tag{50}$$

where A is the location parameter, B is the scale parameter and C is the shape parameter. For such a random variable, the moments are given by

$Mean = A + \frac{C}{B}$
$Variance = \frac{C}{B^2}$
$Skewness = \frac{2}{\sqrt{C}}$
$Kurtosis = A + B(C-1)$

As it can be seen, a two-parameter Gamma and a three-parameter Gamma differ only in the first moment. In our case, $B = \frac{\lambda}{t^H}$, $C = \alpha$ and by setting $A = \frac{-\alpha}{\lambda}t^H$ we achieve a zero mean three parameter Gamma random variable, with $\mathbf{E}[Z] = 0$ and $\mathbf{E}[Z^2] = \frac{\alpha}{\lambda^2}t^{2H}$. Also, by matching moments with the random process $Y(t) = \mu_d t + Z(t)$, which must be $\mathbf{E}[Y(t)] = \mu_d t$, and $\mathbf{Var}[Y(t)] = \sigma^2 t^{2H}$, we find $\mu_d = \frac{\alpha}{\lambda}$ and $\sigma^2 = \frac{\alpha}{\lambda^2}$.

According to [16, pp. 122], if $Y \triangleq aX$, then

$$f_Y(t) = \frac{1}{|a|} f_X\left(\frac{t}{a}\right) \tag{51}$$

A fractal random variable scales in time following the law

$$Z(t) \triangleq t^H Z(1) \tag{52}$$

then

$$f_{Z(1)}(z) = t^H f_{Z(t)}(t^H z) \tag{53}$$

since t is always positive. Simple algrebra shows that

$$t^H f_{Z(t)}(t^H z) = \frac{\lambda}{\Gamma(\alpha)} \left(\lambda \left(z + \frac{\alpha}{\lambda}\right)\right)^{(\alpha-1)} e^{-\lambda(z+\frac{\alpha}{\lambda})} = f_{Z(1)}(z) \tag{54}$$

which proves the fractal scaling behavior of the three-parameter Gamma random variable.

9. APPENDIX B. CORRELATION STRUCTURE OF THE GFN MODEL

As proved in Appendix A $Z(t)$ is self-similar. Although self-similarity should imply long-range dependence in most of the cases, there are some exceptions. One example is Brownian motion, which is $\frac{1}{2}$-sssi (self-similar with stationary increments) with white Gaussian noise as its increment process. The latter not being long-range dependent. On the other hand, there are also some long-range dependent models which are not self-similar, fractional ARIMA being one of them. We must now answer the question: is the Gamma Fractal Noise model long-range dependent?. We shall show that the stationary increments assumption guarantees long-range dependence for the increment process, as noted in [17] without proof.

Remembering that $Y(t) = \mu t + Z(t)$, with $\mathbf{E}[Z(t)] = 0$ and $\mathbf{E}[Z^2(t)] = \sigma^2 t^{2H}$. We can see that

$$\mathbf{E}[Y(t)] = \mathbf{E}[\mu t] + \mathbf{E}[Z(t)] = \mu t \tag{55}$$

$$\mathbf{E}[Y^2(t)] = \mathbf{E}[(\mu t + Z(t))^2] = \mu^2 t^2 + \sigma^2 t^{2H} \tag{56}$$

$$\mathbf{E}[Y(t_2) - Y(t_1)] = \mathbf{E}[\mu|t_2 - t_1| + Z(t_2) - Z(t_1)] = \mu|t_2 - t_1|. \tag{57}$$

Also, by the stationary increments property

$$\mathbf{E}[(Y(t_2) - Y(t_1))^2] = \mathbf{E}[Y^2(t_2 - t_1)] \tag{58}$$

$$= \mu^2|t_2 - t_1|^2 + \sigma^2|t_2 - t_1|^{2H}. \tag{59}$$

But

$$\mathbf{E}[(Y(t_2) - Y(t_1))^2] = \mathbf{E}[Y^2(t_2)] + \mathbf{E}[Y^2(t_1)] - 2\mathbf{E}[Y(t_2)Y(t_1)], \tag{60}$$

thus

$$\mathbf{E}[Y(t_2)Y(t_1)] = \frac{1}{2}(\mathbf{E}[Y^2(t_2)] - \mathbf{E}[(Y(t_2) - Y(t_1))^2] + \mathbf{E}[Y^2(t_1)]) \tag{61}$$

$$= \mu^2 t_2 t_1 + \frac{\sigma^2}{2}(t_2^{2H} - |t_2 - t_1|^{2H} + t_1^{2H}). \tag{62}$$

Also since

$$\mathbf{E}[Y(t_2)Y(t_1)] = \mathbf{E}[(\mu t_2 + Z(t_2))(\mu t_1 + Z(t_1))] \tag{63}$$

$$= \mu^2 t_2 t_1 + \mathbf{E}[Z(t_2)Z(t_1)], \tag{64}$$

we gather that

$$\mathbf{E}[Z(t_2)Z(t_1)] = \frac{\sigma^2}{2}(t_2^{2H} - |t_2 - t_1|^{2H} + t_1^{2H}). \tag{65}$$

We now proceed to find the covariance between the sizes of the video frames fetched from disk, characterized by the Gamma Fractal Noise increment process. Remembering that $X(t) = Y(t) - Y(t-1)$, we have that $\mathbf{E}[X(t)] = \mathbf{E}[Y(t) - Y(t-1)] = \mu$.

The covariance can be written as

$$\begin{aligned}
\mathbf{Cov}[X(i), X(i+k)] &= \mathbf{Cov}[X(1), X(k+1)] && (66)\\
\mathbf{Cov}[X(1), X(k+1)] &= \mathbf{E}[(X(1) - \mathbf{E}[X(1)])(X(k+1) - \mathbf{E}[X(k+1)])] && (67)\\
&= \mathbf{E}[X(1), X(k+1)] - \mu^2 && (68)\\
&= \mathbf{E}[Y(1), Y(k+1)] - \mathbf{E}[Y(1), Y(k)] - \mu^2 && (69)\\
&= \mathbf{E}[Z(1), Z(k+1)] - \mathbf{E}[Z(1), Z(k)] && (70)\\
&= \frac{\sigma^2}{2}((k+1)^{2H} - 2k^{2H} + (k-1)^{2H}). && (71)
\end{aligned}$$

Due to the equivalence between the second difference operator and the derivative, it holds

$$\begin{aligned}
\mathbf{Cov}[X(i), X(i+k)] &= \frac{\sigma^2}{2}\Delta^2 k^{2H} && (72)\\
&\approx \sigma^2 H(2H-1)k^{2H-2} && (73)
\end{aligned}$$

If $\frac{1}{2} \leq H \leq 1$ the autocorrelation coefficient $r(k) = \frac{\mathbf{Cov}[X(i), X(i+k)]}{\sigma^2}$ decays hyperbolically, and is thus, not summable. This proves that the Gamma Fractal Noise is long-range dependent, being this proof valid for any distribution having the first and second order statistics stated above.