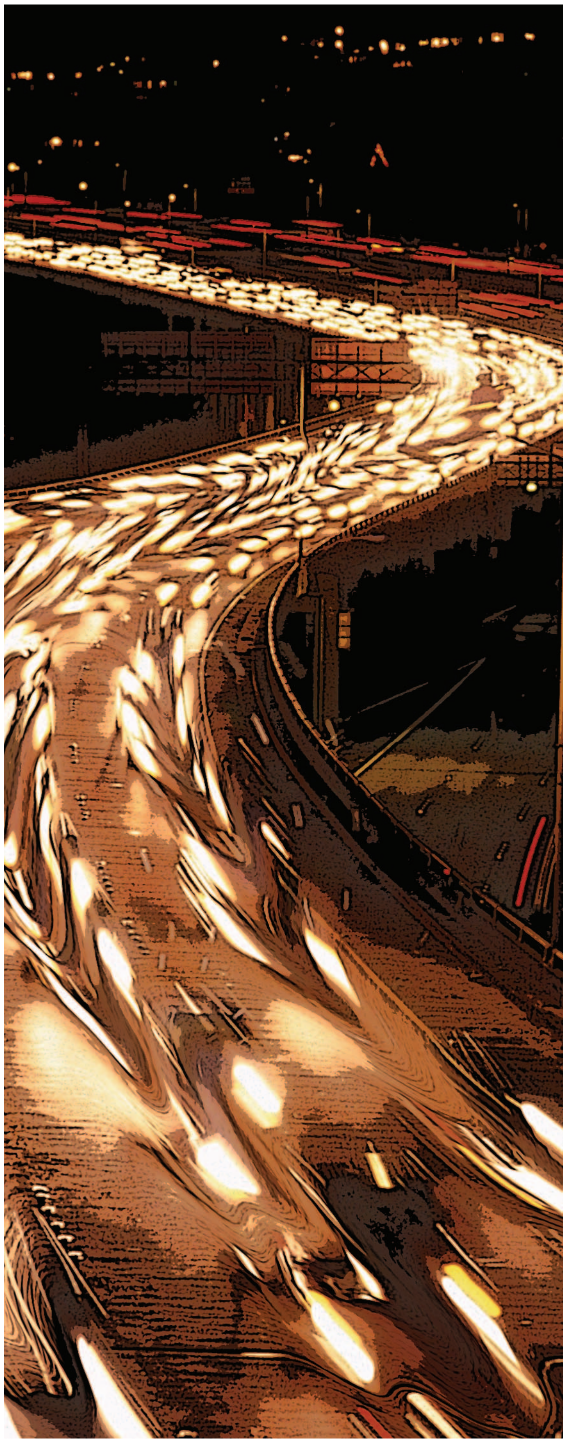
Evaluating multimedia networking mechanisms using video traces

PATRICK SEELING AND MARTIN REISSLEIN


©PHOTODISC

With the increasing popularity of networked multimedia applications, video data is expected to account for a large portion of the traffic in the Internet of the future and in next-generation wireless systems. For transport over networks, video is typically encoded (i.e., compressed) to reduce bandwidth requirements. Even compressed video, however, requires large bandwidths on the order of 100 kb/s or Mb/s. In addition, compressed video streams typically exhibit highly variable bit rates (VBRs). This, in conjunction with the stringent quality of service (QoS) requirements (loss and delay) of video traffic, makes the transport of video traffic over communication networks a challenging problem. As a consequence, in the last decade, networking research on all aspects of video transport has exploded. The characteristics of video traffic, video traffic modeling, as well as protocols and mechanisms for the efficient transport of video streams have received a great deal of interest among networking researchers and network operators.

Video traces, which give the sizes of the individual video frames in a video sequence, have been emerging as convenient video characterizations for networking studies. This article introduces video traces and outlines how they characterize encoded video and can be used in evaluating multimedia networking mechanisms.

Overview of digital video and video coding

Digital video consists of video frames (images) that are displayed at a prescribed *frame rate*; a video frame rate of 30 frames/s is used by the National Television Standards Committee (NTSC). The reciprocal of the frame rate gives the display time of a frame on the screen and is commonly referred to as *frame period*, denoted by T; note that for NTSC video $T = 1/30\,\text{s} = 33.333$ ms. Each individual video frame consists of picture elements (usually referred to as pixels or pels). The *frame format* specifies the size of the individual frames in terms of pixels. The ITU-R/CCIR-601 format (the common TV format) has 720×480 pixels (i.e., 720 pixels in the horizontal direction and 480 pixels in the vertical direction), while the common intermediate format (CIF) format has 352×288 pixels, and the quarter CIF (QCIF) format has 176×144 pixels; the CIF and QCIF formats are typically considered in network-related studies. Each pixel is represented by three components (samples): the luminance component (Y) and the two chrominance components hue (U) and intensity (V), whereby each sample is typically quantized into 8 b. The resulting bit rates are very large. For instance, in a monochrome (black-and-white) CIF video, there are 352×288 Y samples in each video frame, resulting in a frame size of 101,376 B for an uncompressed video frame and a corresponding bit rate of $101{,}376 \times 8$ b/33.333 ms = 24.3 Mb/s, which underscores the need for compression.

Most commercial video codecs, such as RealVideo and WindowsMedia, are derived from the MPEG and H.26x video compression (coding) standards. The two main principles in MPEG and H.26x video coding are

- intraframe coding using the discrete cosine transform (DCT)
- interframe coding using motion estimation and compensation between successive video frames.

In intraframe coding each video frame is divided into blocks of 8×8 samples. Each block is transformed using the DCT into a block of 8×8 transform coefficients, which represent the spatial frequency components in the original block. Typically, the video frame information is concentrated in a few low spatial frequency components, which allows for a more compact representation of the video frame. The transform coefficients are then quantized, whereby the level of coarseness of the quantization is controlled by setting a quantization step size (quantization scale). A larger quantization scale gives a coarser quantization, resulting in a smaller size

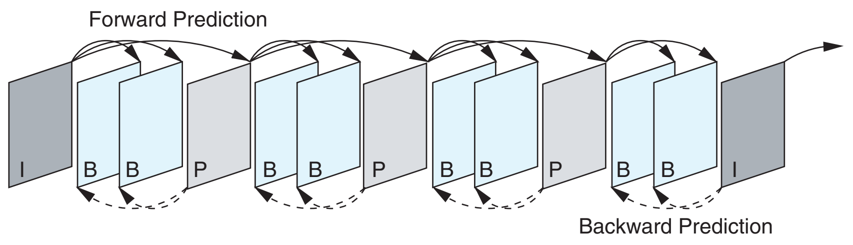


Fig. 1 A typical MPEG GoP pattern with references used for predictive coding of P and B frames

(in bits) of the encoded video frame (and, thus, more compression) but also in a coarser, lower quality video. With *constant bit rate (CBR)* encoding, the quantization scale is dynamically adjusted to keep the bit rate at a prescribed level. With *VBR* encoding, on the other hand, the quantization scale is kept constant, resulting in essentially constant video quality but highly VBRs, as discussed shortly. Since VBR encoding provides consistent video quality and can achieve more efficient compression, it is the focus of much of the multimedia networking research and is also the focus of this article.

For interframe coding, MPEG introduced the frame types intracoded (I), intercoded (P), and bidirectional coded (B). These different frame types are organized into so-called *groups of pictures (GoPs)*. More specifically, the sequence of frames from a given I frame up to and including the frame preceding the next I frame is referred to as one GoP. The pattern of I, P, and B frames that make up a GoP is commonly referred to as *GoP pattern* or *GoP structure*, as illustrated for a typical GoP pattern with three P frames in a GoP and two B frames before and after each P frame in Fig. 1. The different frame types are encoded as follows. In an I frame, all blocks are intracoded as outlined above. P frames, are intercoded (as explained shortly) with reference to the preceding I or P frame, i.e., the preceding I or P frame serves for forward prediction as illustrated by the solid arrows in Fig. 1. B frames, on the other hand, are intercoded with reference to the preceding I or P frame, which serves for forward prediction, and the succeeding I or P frame, which serves for backward prediction, as illustrated by the dashed arrows in Fig. 1.

Table 1. First 16 lines of a trace of an encoding of *Star Wars Episode IV* with a 12 frame GoP pattern. The trace excerpt gives the frame sizes $X_1, X_2, \ldots, X_{16}$ in bits.

1	I	40,488
2	B	18,200
3	B	18,472
4	P	25,584
5	B	19,640
6	B	19,616
7	P	24,528
8	B	18,512
9	B	18,528
10	P	24,880
11	B	20,448
12	B	20,088
13	I	42,544
14	B	18,376
15	B	17,928
16	P	26,144

To intercode a frame, for each block the best matching block in the reference frame(s) is determined and identified by a motion vector. This process is commonly referred to as *motion estimation*. Any (typically small) difference between the block to be encoded and the best matching block is transformed using the DCT and quantized as outlined above. This process is commonly referred to as *motion compensation*. If a good match can not be found in the reference frame(s), the block is intracoded.

Different types of video characterization for network performance evaluation

Generally, there are three different ways to characterize encoded video for the purpose of network performance evaluation:
- video bit stream
- video traffic trace
- video traffic model.

The video bit stream is the actual output of the video encoding and contains the complete video information. One advantage of the bit stream is that it allows for networking experiments where the quality of the video—after suffering losses in the network—can be visually evaluated. One limitation of the bit stream is that it is very large in size: up to several gigabytes for one hour of compressed video or several tens of gigabytes for one hour of uncompressed video. Another limitation of bit streams is that they are usually proprietary and/or protected by copyright. This limits the access of networking researchers to bit streams and also limits the exchange of bit streams among research groups.

Video traces are an alternative to bit streams. While bit streams give the actual bits carrying the video information, traces only give the *number* of bits used for the encoding of the individual video frames. More specifically, let X_n, $n = 1, \ldots, N$, denote the frame size (number of bits) of the encoded (compressed) video frame n, whereby N denotes the number of frames in the video. A video trace typically gives the frame index (number) n, the frame type (I, P, or B), and the frame size X_n (and possibly more detailed information, such as information about video quality metrics) in an ASCII file with one line per frame, as illustrated in Table 1 for the first 16 frames of an encoding of *Star Wars Episode IV*. Since the video traces give only the number of bits representing the video, instead of the actual bits, there are generally no copyright issues.

Video traffic models strive to capture the essential properties of the real traffic in an accurate, computationally efficient, and preferably mathematically tractable description, which should also be parsimonious, i.e., require only a small number of parameters. A traffic model is typically developed based on the statistical properties of a set of video trace samples of the real video traffic. The developed traffic model is verified by comparing the traffic it generates with the video traces. If the traffic model is deemed

Table 2. Traffic statistics for QCIF videos with $N = 108{,}000$ frames encoded with MPEG-4 with fixed quantization scales corresponding to a high video quality.

Video Title	Frame Size Stats			GoP Size Stats	
	Mean $\bar{X}$ [kB]	CoV CoV_x	Peak/M $X_{max}/\bar{X}$	CoV CoV_y	Peak/M $Y_{max}/\bar{Y}$
Star Wars Episode IV	2.30	0.61	7.61	0.38	5.05
Silence of the Lambs	1.88	0.77	8.72	0.59	7.37
The Tonight Show	2.74	0.88	7.23	0.71	4.56

sufficiently accurate, it can be used for the mathematical analysis of networks, for model-driven simulations, and also for generating so-called virtual (synthetic) video traces.

Video traffic statistics

Video traffic is typically characterized with elementary statistics of frame sizes X_n, $n = 1, \ldots, N$. Commonly, the average (arithmetic mean) of the frame size

$$\bar{X} = 1/N \times \sum_{n=1}^{N} X_n,$$

the coefficient of variation CoV_X, defined as the standard deviation of the frame size normalized by the average frame size, i.e., $\mathrm{CoV}_X = \sigma_X / \bar{X}$ with

$$\sigma_X^2 = 1/(N-1) \times \sum_{n=1}^{N} \left[X_n - \bar{X}\right]^2,$$

and the peak-to-mean ratio of the frame size $X_{\max}/\bar{X}$, where $X_{\max}$ denotes the size of the largest frame in the trace, are considered. These frame size statistics are given in Table 2 for one-hour segments (N = 108,000) of MPEG-4 encodings of popular movies and a late-night TV show. Note that the bit rates corresponding to the frame sizes reported in Table 2 are obtained by dividing the frame sizes by the frame period, e.g., the mean bit rate of the *Silence of the Lambs* video is $\bar{X}/T$ = 1.88 kb/33.333 ms = 564 kb/s. The values for the CoV and the peak-to-mean ratio of the frame sizes indicate that the video traffic is highly variable. For the *Silence of the Lambs* encoding, the largest frame is 8.72 times larger than the average frame size; correspondingly, the largest (peak) bit rate of the video is 8.72 × 564 kb/s = 4.92 Mb/s. A significant part of this variability of the frame sizes is due to the frame size variations within one GoP. As illustrated in Table 1, P frames that are encoded with forward prediction are typically smaller than the intracoded I frames; the B frames that are encoded with forward and backward prediction in turn are typically smaller than the P frames.

To remove the frame size variation within the individual GoPs from consideration and allow the study of the underlying variations in the video traffic, the sizes of the frames in each GoP are summed up to obtain the *GoP sizes* (in bits). Let Y_m, $m = 1, \ldots, M$, denote the size of the mth GoP in the video, and note that a video consisting of N frames and encoded with a GoP pattern consisting of 12 frames has $M = N/12$ GoPs, i.e., each of the video encodings in Table 2 has $M = 108,000/12 = 9,000$ GoPs. The coefficient of variation CoV_Y and the peak-to-mean ratios $Y_{\max}/\bar{Y}$ of the GoP sizes, defined analogously as for the frame sizes and reported in Table 2, indicate that the GoP sizes are significantly less variable than the frame sizes. Nevertheless, there are still quite large variations in the GoP sizes, as also illustrated by the plot of the GoP sizes Y_m as a function of the index m in Fig. 2(a). These variations are largely due to the different scene contents in the videos. Scenes with a lot of detail and/or a high level of motion are less amenable to compression (encoding) and result, therefore, in larger GoP sizes. For instance, there is a very detailed high motion scene at the very beginning of the movie, and also a detailed high motion segment from approximately GoP number 600 to 1,000.

The variability of the video traffic poses challenges for the allocation of network resources, such as link buffers and bandwidth, for video traffic. When resources are allocated according to peak bit rate, the network will be underutilized most of the time. On the other hand, when resources are allocated according to average bit rate, the link bandwidth will be insufficient to accommodate the video traffic during the periods of higher than average bit rate traffic, resulting in large backlogs of traffic in the buffers preceding the links. This problem is further exacerbated by the typical characteristic of encoded video to have persistent periods (bursts) of higher than average bit rate traffic. This persistency of the traffic bursts is related to the scene structure of the video, since the traffic rate typically stays within a certain range in each scene. One way to measure the persistency of the traffic bursts is to plot the autocorrelation function of the sequence of GoP sizes $Y_1, Y_2, \ldots, Y_M$, which is defined for a lag of k GoPs as

$$\rho(k) = \frac{1}{M-k} \sum_{m=1}^{M-k} \frac{\left(Y_m - \bar{Y}\right)\left(Y_{m+k} - \bar{Y}\right)}{\sigma_Y^2}$$

and is plotted for *Star Wars Episode IV* in Fig. 2(b). The slower the autocorrelation function drops off, the stronger the correlations and the more persistent the bursts of traffic. The plotted autocorrelation function decays relatively slowly for a lag of 80 GoPs, which corresponds to approximately 32 s, the autocorrelation coefficient is still about 0.2. This indicates fairly significant correlations

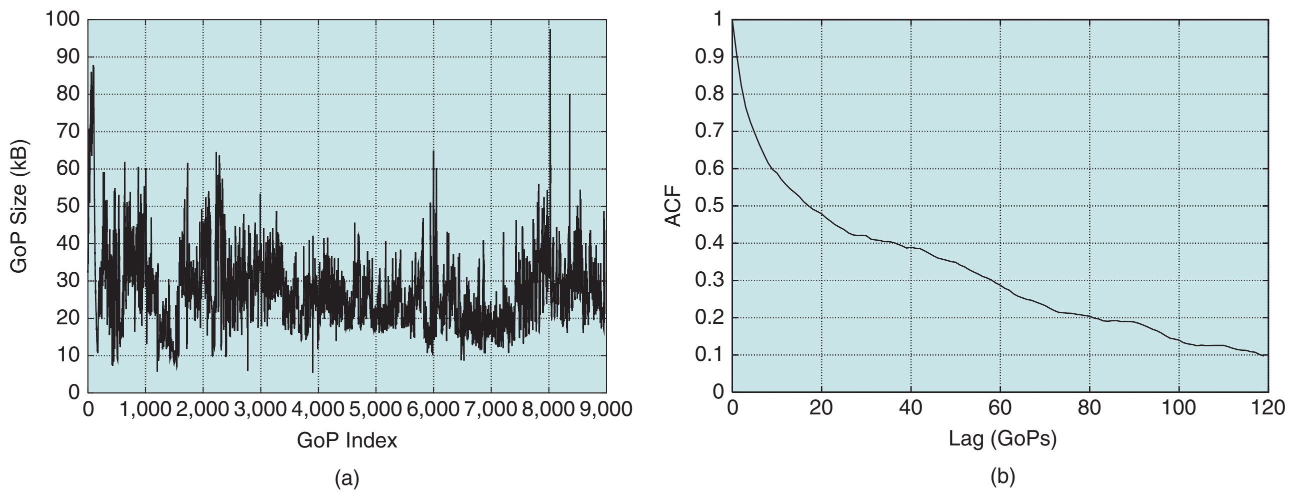


Fig. 2 Characteristics of GoP sizes of *Star Wars Episode IV* encoding: (a) GoP size as function of the GoP index and (b) autocorrelation function

over relatively long time periods, which exacerbate the problem of backlog building up during traffic bursts and possibly leading to buffer overflows and the loss of video data.

These challenges in transporting video traffic over communication networks, such as the Internet, combined with the tremendous increase in popularity of multimedia Internet applications is fueling the development of more and more sophisticated video traffic management mechanisms. These mechanisms attempt to judiciously exploit network resources (buffer, bandwidth) as well as the client buffer and to time the transmission of the video data so as to provide good video quality with a bandwidth only slightly above the average bit rate. An important component of the research on traffic mechanisms is the evaluation of their performance, which is often done by simulating the operation of the mechanism with video traces as described next.

Using video traces for evaluating networking mechanisms: Performance metrics

There are three broad areas that require careful consideration when using video traces in simulations, namely

• the definition of the video-related performance metrics

• the generation of the video traffic work load for the system under study

• the statistically sound estimation of the performance metrics of interest.

Simulations with video traces can be used to evaluate conventional network performance metrics, such as the utilization of the networking resources, delay, and buffer overflow probabilities. In addition, simulations with video traces can be used to evaluate performance metrics that are related to the video, such as the starvation probability, which gives some indication of the quality of the video delivery. The starvation (loss) probability comes in two main forms. The *frame starvation probability* is the long run fraction of video frames that miss their decoding (playout) deadline, i.e., they are not completely delivered to the receiver by the time the receiver needs them to start the decoding. The *information-loss probability* is the long run fraction of encoding information (bits) that misses its decoding (playout) deadline. The information-loss probability has a finer granularity than the frame-loss probability because a partially delivered frame is considered as one lost frame toward the frame-loss probability (irrespective of how much of the frame was delivered/ not delivered in time), whereas the information-loss probability counts only the fraction of the frame's information bits that were not delivered in time. As an illustrative example, consider the transmission of 10 240-b frames to a client and suppose only 120 b of the first frame are delivered on time (and the other 120 b arrive after the decoding deadline). Also, suppose the remaining nine frames are all completely delivered ahead of their respective decoding deadlines. Then the frame-loss probability is $1/10 = 10\%$, whereas the information-loss probability is $120/(10 \cdot 240) = 5\%$.

Although the frame- and information-loss probabilities are convenient and widely used performance metrics for video networking, they have a number of limitations. One limitation is that the loss probabilities ignore the dependencies between the encoded video frames. Specifically, in an MPEG encoding, the I frame in a GoP is required to decode all other P and B frames in the GoP. Thus, the loss of an I frame affects all the frames in the GoP. Another shortcoming is that the loss probabilities provide only limited insight into the visual video quality perceived by the user. A smaller loss probability corresponds in general to a higher video quality, however, quantifying this relationship is very difficult and the topic of ongoing research. Overall, the assessment of the visual quality of received video after network transport is a very challenging active research area. The starvation (loss) probabilities are, therefore, commonly employed in evaluating the performance of multimedia networking mechanisms.

Generating video traffic workload from traces

When generating the video traffic workload, there is a range of issues to consider, from choosing and preparing the video streams (traces) to running the actual simulations. The specific simulation approach used in an evaluation depends to a large degree on the specific networking scenario and mechanism under study. For an illustration of a typical simulation design, we outline the so-called *constant utilization simulation scenario*. This simulation scenario is suitable for evaluating the performance of a multiplexer, scheduler, or similar network system that is fed by several streams with a specific long run average utilization level.

The first consideration is typically to select the videos (titles) and to select and prepare the corresponding video traces. It is advisable to select as many different videos as available in video trace libraries (as for instance in the library at <http://trace.eas.asu.edu>) from the video genre(s) that will be transported over the network. Let M denote the number of different videos selected for a given evaluation study.

The next step is to select and prepare the traces for the selected M videos. For the constant utilization simulation, it is typically desired that all videos have the same average bit rate r (in b/s). This common desired bit rate r is achieved as follows. For each video, there are typically traces for encodings with different quantization scales and correspondingly different average bit rates available in a video trace library. First, pick the trace of the encoding with the average bit rate $\bar{X}/T$ that is closest to the desired bit rate r. Second, multiply each frame size in the picked video trace by the factor $r/(\bar{X}/T)$.

Next, determine the number of simultaneous video streams required in the simulation to achieve the desired level of system utilization, say a utilization level μ. Noting that the long run average utilization of the system is given by $\mu = J \cdot r/C$, where C denotes the capacity of the system (in b/s), we can solve for the number of required streams as $J = \mu \cdot C/r$.

Having selected and prepared the video traces and determined the number of simultaneous streams to be simulated, we can move on to the actual simulation of the network mechanism under study, which proceeds as follows. For each of the J video streams, one of the M traces is uniformly randomly selected, i.e., each trace is equally likely selected with probability $1/M$ to satisfy a client request. For each selected trace, a starting (frame) phase is independently drawn from a discrete uniform distribution over the N frames in the trace. The video frames are then processed according to the networking mechanism under study from the starting frame onward.

The next question that arises is for how long, i.e., for how many frames should the mechanism under study be simulated? One option is to continue the simulation for N frames, i.e., for the full length of the traces. (Note that, due to the random starting frame, the end of the traces may be reached before processing all N frames. When the end of a trace is reached the trace is "wrapped around," i.e., the processing continues from the beginning of the trace.) Once all N frames have been processed, we immediately randomly select a new trace and starting phase into the trace for each of the J streams. Thus there are always J streams in progress. Another option is to not continue the simulation until all N frames of a trace have been processed but instead to draw a random independent stream duration L_j frames (which is bounded by N) for each individual stream j, $j = 1, \ldots, J$. Whenever all L_j frames of a stream j have been processed, a new trace, starting phase, and duration L_j are randomly drawn for the stream immediately.

Estimating performance metrics

As with any simulation, a key consideration when simulating a network mechanism using video traces is the statistical validity of the obtained results. Generally, in simulation studies either a terminating simulation approach or a steady-state simulation approach is used to obtain statistically valid results. Both approaches can be applied to simulations with video traces, as discussed in the following.

In terminating simulations, several independent simulation runs are performed and the estimates of the metrics of interest are obtained by averaging the metric samples obtained from the individual runs. A terminating simulation of the constant utilization scenario can be conducted by running several simulations for a fixed number of video frames, e.g., all the N frames in the videos as outlined above. Each simulation is started with independently randomly selected traces and starting phases. Each simulation for a set of video traces and starting phases gives one sample of the metrics of interest (e.g., information-loss probability). The advantage of this terminating simulation approach is that the individual simulation runs are independent, thus, the classical student t or normal distribution based statistics can be used to evaluate the confidence intervals around the means of the collected samples.

The disadvantage of the terminating simulation approach is that each simulation run needs to be "warmed up" sufficiently to remove the initial transient (i.e., the metrics of interest are not observed during the warm-up period but only after the system has reached its steady-state operating region). While this is not a problem for system simulations, which do not require any warm-up, e.g., the simulation of a bufferless multiplexer for a constant utilization, the warm-up may be a significant problem for systems that need warm-up, e.g., buffered multiplexers. Determining the sufficiently long warm-up period for simulations with video traces is largely an open research problem. It is, therefore, advisable to conservatively warm up the system for relatively long periods and to inspect whether the system has reached that steady-state operating region before observing the metrics of interest.

With the steady-state simulation approach, a single (typically very long) simulation run is performed, and the metrics of interest are typically obtained by averaging metric samples obtained during independent observation periods (usually referred to as batches). A steady-state simulation with video traces can be conducted by running one long constant utilization simulation. The advantage of the steady-state simulation is that the warm-up period is incurred only once. The challenge of the steady-state simulation of systems with video traces is that, due to the correlations in video traffic over long time periods, the metric estimates of successive (nonoverlapping) observation periods (batches) are typically somewhat correlated. A simple heuristic to obtain uncorrelated batches, which are required to apply the standard normal distribution-based confidence interval calculation, is to separate successive observation periods (batches) such that they are (approximately) independent. More specifically, the heuristic is to run the constant utilization and to truncate the distribution of the stream duration at a specific value $L_{\max}$ frames. Then, separating successive batches by at least $L_{\max}$ frames will ensure that none of the video streams that contribute to the traffic load during a given batch contributes to the traffic load during the next batch. This ensures that the successive batches are approximately independent. This heuristic provides a simple way to obtain statistically meaningful performance metrics at the expense of increased simulation duration.

Future directions in network performance evaluation with video traces

Ongoing developments of video traces are focused on providing video traces for evaluating emerging video services, such as Internet protocol based (IP) television, a.k.a. IPTV, and high-definition television (HDTV) over the Internet. Also, advanced video traces are being developed that allow for accurate estimates of the visual quality of the received video after lossy network transport.

Read more about it

• M. Dai and D. Loguinov, "Analysis and modeling of H.264 and MPEG-4 multi-layer video traffic," in *Proc. IEEE Infocom*, Miami, FL, 2005, pp. 2257–2267.

• F. Fitzek and M. Reisslein, "MPEG-4 and H.263 video traces for network performance evaluation," *IEEE Network*, vol. 15, pp. 40–54, Nov./Dec. 2001.

• A.M. Law and W.D. Kelton, *Simulation, Modeling and Analysis*, 3rd ed. New York: McGraw Hill, 2000.

• P. Seeling, M. Reisslein, and B. Kulapala, "Network performance evaluation with frame size and quality traces of single-layer and two-layer video: A tutorial," *IEEE Commun. Surveys and Tutorials*, vol. 6, pp. 58–78, 3rd qtr, 2004.

• U.K. Sarkar, S. Ramakrishnan, and D. Sarkar, "Modeling full-length video using markov-modulated gamma-based framework," *IEEE/ACM Trans. Networking*, vol. 11, pp. 638–649, Jul. 2003.

About the authors

Patrick Seeling received the Dipl.-Ing. degree in industrial engineering and management (specializing in electrical engineering) from the Technical University of Berlin (TUB), Germany, in 2002. Since 2003, he has been a Ph.D. student in the Department of Electrical Engineering at Arizona State University. His research interests are in the area of video communications in wired and wireless networks. He is a student member of the IEEE and the ACM and can be reached at patrick.seeling@asu.edu.

Martin Reisslein is an assistant professor in the Department of Electrical Engineering at Arizona State University, Tempe. He received his Ph.D. in systems engineering from the University of Pennsylvania, Philadelphia, in 1998. He maintains an extensive video trace library at <http://trace.eas.asu.edu>. He is editor-in-chief of the *IEEE Communications Surveys and Tutorials* and can be reached at reisslein@asu.edu.

This article is an adapted version of 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," *IEEE Commun. Surveys and Tutorials*, vol. 6, pp. 58–78, 3rd qtr, 2004.