Hindawi Publishing Corporation
Advances in Multimedia
Volume 2012, Article ID 539396, 13 pages
doi:10.1155/2012/539396

Research Article

High-Definition Video Streams Analysis, Modeling, and Prediction

Abdel-Karim Al-Tamimi,[1] Raj Jain,[2] and Chakchai So-In[3]

[1] *Computer Engineering Department, Yarmouk University, Irbid 21163, Jordan*
[2] *Department of Computer Science and Engineering, Washington University in St. Louis, St. Louis, MO 63130, USA*
[3] *Department of Computer Science, Khon Kaen University, Khon Kaen 4002, Thailand*

Correspondence should be addressed to Abdel-Karim Al-Tamimi, altamimi@yu.edu.jo

Received 26 November 2011; Revised 7 February 2012; Accepted 7 February 2012

Academic Editor: Marios C. Angelides

High-definition video streams' unique statistical characteristics and their high bandwidth requirements are considered to be a challenge in both network scheduling and resource allocation fields. In this paper, we introduce an innovative way to model and predict high-definition (HD) video traces encoded with H.264/AVC encoding standard. Our results are based on our compilation of over 50 HD video traces. We show that our model, simplified seasonal ARIMA (SAM), provides an accurate representation for HD videos, and it provides significant improvements in prediction accuracy. Such accuracy is vital to provide better dynamic resource allocation for video traffic. In addition, we provide a statistical analysis of HD videos, including both factor and cluster analysis to support a better understanding of video stream workload characteristics and their impact on network traffic. We discuss our methodology to collect and encode our collection of HD video traces. Our video collection, results, and tools are available for the research community.

1. Introduction

Web-based video streaming websites facilitate the creation and distribution of digital video contents to millions of people. Websites like YouTube [1] are now considered to be among the most accessed websites by Internet users. Such websites are now accounting for 27 percent of the Internet traffic, rising from 13 percent in one year [2]. Internet video traffic is expected to amount to 50% of consumer Internet traffic in 2012 [3].

This surge in traffic percentage can be explained by the latest surveys that show that the percentage of US Internet users watching streaming videos has increased from 81% to 84.4%, and the average time spent per month increased from 8.3 to 10.8 hours/month in just three months period July–October of 2009 [4, 5]. Additionally, several websites, for example, Hulu [6] and Netflix [7], have started offering access to TV shows and selected movies that has increased the reliance of the daily Internet users on such websites and augmented their expectations of the level of services and quality of delivery.

Resource and bandwidth allocation schemes for video streaming are dependent on their ability to predict and manage the time variant demand of video streams. Existing dynamic resource allocation schemes [8–10] utilize video traffic prediction to offer better accommodation for existing video traffic, and allow higher admission rates. The traffic predictor is the most important part in dynamic bandwidth allocation. It is can be based either on traffic characteristics or on the video content. Video-content-based traffic predictors have shown their superiority over their traffic-based counterparts [10].

Therefore, it essential to analyze and model video traffic to allow better quality of service (QoS) support. In this paper, we present the results of our model-based predictor and discuss its video traffic prediction capabilities.

Modeling video streams is a challenging task because of the high variability of video frame sizes. Such variability has

increased with the introduction of MPEG4-Part10/advanced video codec (AVC)/H.264 high-definition video codec standard. AVC provides better compression rate (i.e., lower mean values) than its predecessors. Yet at the same time, it results in higher frame size variability [11].

In this paper, we present our work to analyze, model, and predict high-definition (HD) video traces encoded with the H.264/AVC codec. We present results based on over 50 HD video traces. We compare three modeling methods: autoregressive (AR) [12], autoregressive integrated moving average (ARIMA) [12] using the approach proposed in [13], and our Simplified Seasonal ARIMA Model (SAM) that was developed for the less resource demanding mobile video traces [14, 15]. In addition we compare these models in their prediction accuracy.

There have been several contributions that aimed to achieve a better understanding of the relationship between the statistical characteristics of video traces and their impact on data networks. In [16], the authors presented a statistical and factor analysis study of 20 MPEG1 encoded video traces and their impact on ATM networks. Similar approaches were presented in [17] with emphasis on video trace frame size distribution. The author in [18] performed a statistical analysis on four MPEG4-AVC encoded video traces demonstrating the quantization effects over several statistical measurements, including the intercorrelation between video frames. In [11], the authors compared the statistical characteristics of AVC standard versus its predecessor, namely, MPEG4-Part2 in terms of bit rate distortion performance, bit rate variability, and long-range dependence (LRD).

In this paper, we present our work of analyzing and modeling over 50 HD video traces from YouTube HD videos section. We aim through this contribution to investigate the main statistical characteristics that define an HD video trace. This identification is important for two main reasons: it helps in clustering video traces depending on certain statistical criteria to help choose the correct traffic workload, or in other possible data mining processes [16]. Additionally, it helps define the main statistical attributes of HD video traces that should be considered to achieve a valid statistical model [19].

One of the main challenges in developing a valid video workload model is the availability of an adequate number of traces to test the proposed model. The available traces on the web are scarce and do not represent all the different types of videos. Thus, one of the aims of this contribution is to provide researchers with a sufficient number of traces to support their future studies. All our tools, results, and video traces are available through our website [20].

In addition to analyzing and modeling these video traces, we provide several tools: a trace generator based on our model that can be used to generate user-defined traces with the desired statistical characteristics, and a simple GUI interface to provide the essential statistical analysis and comparison graphs for HD video traces. The trace generator can also be used to produce a new movie trace that represents a blend of different video characteristics. Figure 1 summarizes the main steps taken in analyzing and modeling the selected videos and shows each step's corresponding outputs.

Our encoding process starts with an HD YouTube video in *mp4* format, which is then converted to a YUV (4 : 2 : 0) raw video format. Such format allows video frames to be much more compressible [21]. The raw video is consequently encoded with AVC, and the process produces the following: an encoded movie file, its encoding statistics file, and a full verbose description of the encoding process. The verbose output is then parsed using our analysis tool to get the video trace information, which is then modeled using AR, ARIMA, or SAM. The video trace is used also to produce the video frames autocorrelation function (ACF) and the partial autocorrelation function (PACF) graphs. ACF plots are commonly used tools to check for randomness in a data series by plotting the data set values over several time lags [22]. Given a data series X_t, PACF for a lag k is the autocorrelation between X_t and X_{t-k} that is not accounted for by lags 1 to $k-1$ inclusive.

The SAM parameters for each video can be used in either video traffic prediction analysis, or in generating video traces. SAM frame generator uses these parameters to generate a movie trace that is statistically close to the original movie trace.

This paper is organized as follows: Section 2 discusses the methodology of obtaining and encoding our collection of HD videos. Section 3 shows the results of our statistical analysis, including both factor and cluster analysis. In Section 4, we compare the results of modeling the video traces and provide a simple introduction to SAM. Section 5 discusses the approach to evaluate the prediction accuracy of the compared models and the comparisons results. Section 6 illustrates our tools design and their implementations. Finally, we conclude the paper and give some insights to the impact of our results and our future work.

2. Encoding YouTube HD Videos

To represent real life video traffic load, we chose YouTube website as our source. YouTube is currently the most popular video streaming website on the Internet [23]. Our first step in selecting the candidate videos from YouTube was to make sure that we have a good variety of both texture/details and motion levels. To select a representative group of the available videos, we started our selection process with 9 videos of the most visited videos in YouTube HD section [1]. Then, we increased our collection by selecting three random videos from each of the 15 subcategories available for YouTube website's users. In total we have collected 54 video files in *mp4* format.

Then, we analyzed the collected videos using *MediaInfo* [24] to determine the encoding parameters for the various videos and to select the most commonly used parameter values. We made sure that the parameter values we selected were consistent with those recommended in [25, 26] for YouTube video encoding. Our next step was to convert all these videos to raw or YUV (4 : 2 : 0) format. This step is important to ensure unified encoding parameters for all the collected videos to allow objective comparisons. We performed the conversion process using the open source coding library FFMPEG [27].

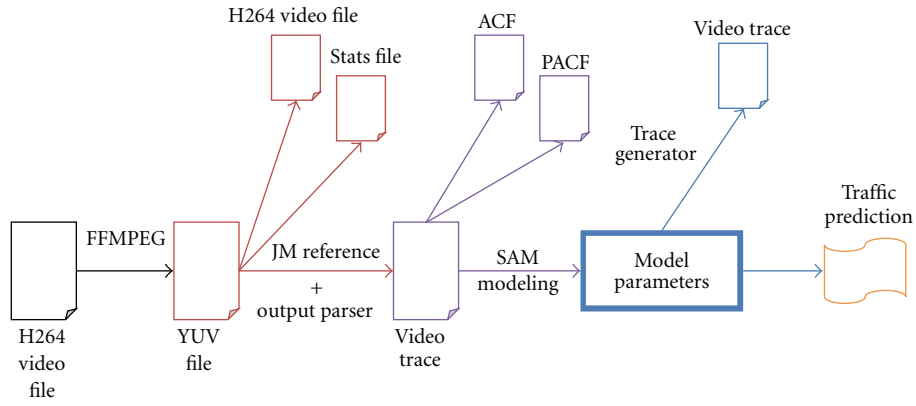


Figure 1: Modeling, analyzing, and generating video traces processes.

Table 1: Encoding parameters for the selected YouTube video collection.

Encoding parameter	Value
FrameRate	24
OutputWidth	1280
OutputHeight	720
ProfileIDC	100 (High)
LevelIDC	40 (62914560 samples/sec)
NumberBFrames	2
IDRPeriod	24
NumberReferenceFrames	3
QP (quantization parameter)	$I = 28, P = 28, B = 30$

To convert YUV files to the H.264/AVC format, we tested two publically available encoding libraries: x264 [28] and JM reference software [29]. Though x264 is significantly faster than JM reference software, it provided us with less information about the encoding process. Table 1 lists the main encoding parameters used with JM reference software.

These parameters were chosen to represent the majority of the videos we have collected. We used in our encoding process Instantaneous Decoding Refresh (IDR) frames [25]. IDR frames are special type of I frames that allow better seeking precision and thus enhance the user's experience. We used *closed-GOP* setting [26] to ensure that all I frames are IDR frames, hence improving the user's online experience. The majority of the collected videos have a frame rate of 24 fps. *IDRPeriod* defines the periodicity of IDR frames.

The *ProfileIDC* parameter defines the video profile, which, in this case, is set to high. This parameter, along with the *LevelIDC* parameter, specifies the capabilities that the client decoder must have in order to decode the video stream. Parameter *NumberBFrames* specifies the number of *B* slices or frames between *I*, IDR, and *P* frames. The quantization parameters used are the default values for the encoder. The parameter *NumberReferenceFrames* sets the maximum number of reference frames stored in the decoder buffer, and it is set to three frames. All other encoding parameters are set to the default values of JM reference software. In the course of our analysis and encoding processes, we used two versions of JM reference software: v15.1 and v16.0.

The encoding procedure is both time and resource consuming process. The encoding of a single video file took on average 37 hours, with an average encoding rate of 0.02 fps. The average size of a raw YUV (4 : 2 : 0) video file is around 4 GB. The encoding was done using a 2.8 GHz Core i7 machine with 6 GB of DDR3 RAM. These figures support our conviction of the necessity to have a valid trace model and generator. The output of the encoding process is then run through our parser to extract the video trace frame size information needed for the next steps of our analysis and modeling.

3. Factor and Cluster Analysis of Video Traces

In this section we discuss the steps taken to perform a full statistical analysis of the collected video traces in order to achieve a better understanding of the main factors that can be used to represent a video trace in order to develop a representative statistical model.

Multivariate analysis is used to reveal the full structure of the collected data, and any hidden patterns and key features [30]. Multivariate analysis is used especially when the variables are closely related to each other, and there is a need to understand the underlying relationship between them. We have computed the following statistical quantitative values for traces frame sizes: mean, minimum, maximum, range, variance, standard deviation, the coefficient of variance, and the median value. In addition, we computed the Hurst exponent value, as shown in (4), which indicates the video sequence's ability to regress to its mean value, with higher values indicating a smoother trend, less volatility, and less roughness. Its value varies between 0 and 1. This is also an indication of the strength of the long-range dependence (LRD) among video frames [19]. The Hurst exponent can be computed by first calculating the mean adjusted series Y:

$$Y_i = x_i - \overline{x}, \quad i = 1, 2, \ldots, N, \tag{1}$$

Table 2: Range of statistical values for the collected video traces.

	Mean	Range	Variance	Hurst	Coefficient of variance	Median	Skewness	Kurtosis
Max	83340.43	1198416	13767760363	0.902836	3.9860815	62748	6.58066	61.34631
Min	9782.01	65576	154362485	0.498937	0.6875022	448	0.2287191	1.643709

where x_i is the frame size at index i, $\bar{x}$ is the mean frame size over the trace length N, then we calculate the cumulative deviate vector S:

$$S_t = \sum_{j=1}^{i} Y_j, \quad i = 1, 2, \ldots, N. \tag{2}$$

The next step is to calculate the range value R, and we divide it over the standard deviation value denoted by σ:

$$R = \frac{\max(S) - \min(S)}{\sigma}, \tag{3}$$

$$\text{Hurst Index} = \frac{\log(R)}{\log(N) - \log(2)}, \tag{4}$$

where x_i is the frame size at index i, $\bar{x}$ is the mean frame size, and N is the number of frames in the trace. We also computed the skewness value that represents the symmetry of the observed distribution around its center point [19]:

$$\text{Skewness} = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^3}{(N-1) \times \text{std}^3}. \tag{5}$$

Here std is the standard deviation of the frames sizes. Additionally, we computed the kurtosis value, which is an indication whether the observed video trace distribution is peaked or flat relative to a normal distribution [19]. The kurtosis equation is illustrated below

$$\text{Kurtosis} = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^4}{(N-1) \times \text{std}^4}. \tag{6}$$

As Table 2 shows, the collected videos represent a statistically diverse data samples. And as we mentioned before, the video frame size variance of HD videos is considerably substantial. The table shows the most important variables that have been collected. We noticed through our preparation analysis that min variable does not contribute to the total variance significantly, and thus it was disregarded. Both max and range, and variance and standard deviation pairs are almost identical. We picked range and variance to represent the two pairs, respectively. In the next subsections, we will discuss the methodology and results of performing both factor and cluster analysis.

3.1. Principal Component Analysis. One of the most common factor analysis methods is principal component analysis (PCA) [16], where a group of possibly related variables are analyzed and then reduced to a smaller number of uncorrelated factors. These factors accounts for most of the variance in the observed variables. By performing this process, we aim to minimize the number of variables to represent a video trace without much loss of information [30].

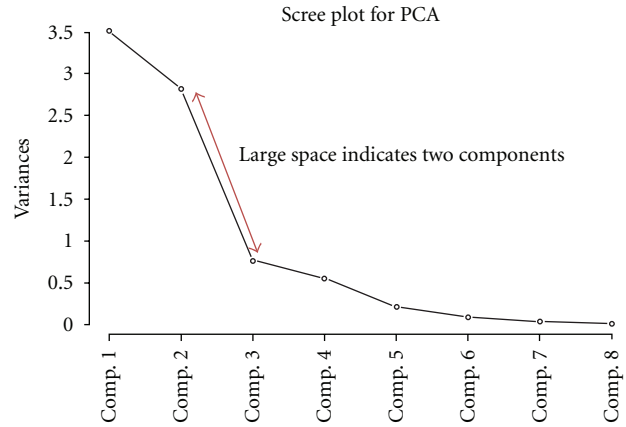


Figure 2: Scree plot for the HD video collection data based on the eight selected variables which indicates two principal components.

Our first step is to determine the smallest number of variables to represent each video trace. Table 3 shows the correlation between the selected variables. These variables collectively represent the majority of the samples variation.

The importance of each factor is represented by its eigenvalue. To determine the number of factors to extract we used Kaiser-Guttman rule [31]. By following this rule, we excluded the factors with eigenvalue less than 1. We supported our selection by performing the Scree test [32] as shown in Figure 2, where we plotted the relationship between the number of factors and their cumulative contribution to the total variance of the data set, and we looked for either large spaces between the plotted variables or a knee in the graph to determine the number of factors to be considered.

Our analysis resulted in choosing two factors with the following eigenvalues: $\lambda_1 = 3.51$, and $\lambda_2 = 2.82$. These factors account for $79\%[(\lambda_1 + \lambda_2)/8]$ of the total standardized variance. We confirmed that the number of factors is sufficient to explain the intercorrelations among variables by performing several nongraphical tests [33].

To simplify the factor structure and spread out the correlations between the variables and the factors (their loadings values) as much as possible, we performed both orthogonal and oblique rotations on the factors [34]. We chose *varimax* orthogonal rotation as it gave the best results. As shown in Figure 3, the two significant groups are the mean and skewness groups. Table 4 shows the loadings values for both varimax rotated and unrotated factors.

As can be noticed, the rotated factors are better spread out and simpler to interpret. From Table 4 we can note that the first factor F_1^* defines mainly mean and variance values.

Table 3: Correlation between the selected variables.

	Mean	Range	Var	Hurst	c.var	Median	Skew	Kurt
Mean	1	0.48	0.73	0.48	−0.40	−0.9	−0.36	−0.23
Range	0.48	1	0.74	0.34	0.19	0.25	0.51	0.6
Var	0.73	0.74	1	0.36	0.13	0.41	0.13	0.14
Hurst	0.48	0.34	0.36	1	−0.44	0.41	0.25	0.17
c.var	−0.40	0.19	0.13	−0.44	1	−0.56	0.71	0.51
Median	−0.9	0.25	0.41	0.41	−0.56	1	−0.49	−0.33
Skew	−0.36	0.51	0.13	0.25	0.71	−0.49	1	0.93
Kurt	−0.23	0.6	0.14	0.17	0.51	−0.33	0.93	1

Table 4: Estimated and rotated factors loadings.

	Estimated		Rotated (varimax)	
	F_1	F_2	F_1^*	F_2^*
Mean	0.84	0.46	**0.93**	—
Range	—	0.95	0.73	0.62
Variance	0.39	0.80	0.84	—
Hurst	0.62	—	0.64	—
C. Var	−0.75	0.35	—	0.77
Median	0.87	—	0.77	−0.46
Skewness	−0.75	0.62	—	**0.97**
Kurtosis	−0.62	0.67	—	**0.91**

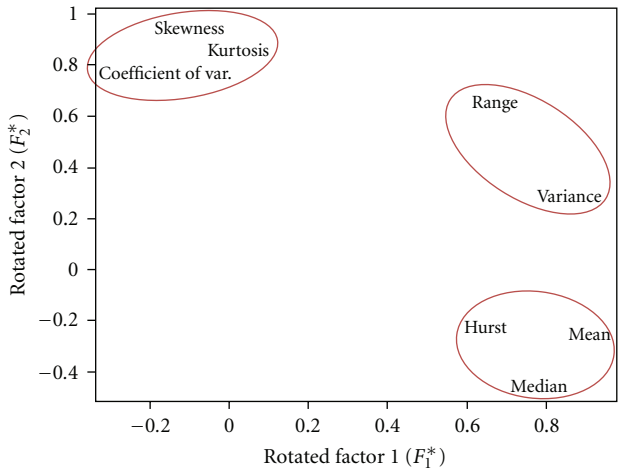


Figure 3: Scatter plot of varimax rotated factors F_1^* and F_2^* in the space of the two principal components.

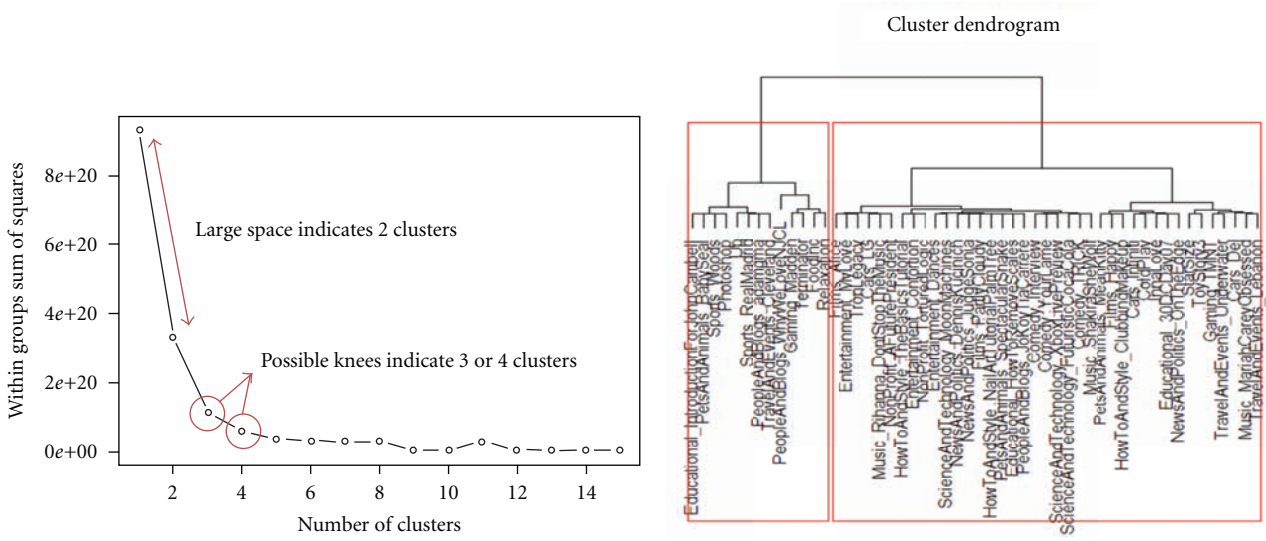
The second factor defines mainly skewness and kurtosis values. We chose *mean* to represent the first factor since it has the highest load. We chose kurtosis as a representative of F_2^* since it has the lowest correlation between it and the mean (−0.23). This analysis shows the importance of skewness and kurtosis in HD videos traces. These two variables were considered irrelevant in previous video analysis [16]. This realization can be explained by the dependence of these variables on the standard deviation that accounts for a significant proportion of the total variance of HD videos traces.

3.2. Cluster Analysis Using k-Means Clustering. We have demonstrated that the selected two factors, or principal components, are sufficient to characterize a HD video trace. The second step of our analysis is to group the collected video traces into clusters. We used one of the most popular clustering methods: *k-means* clustering algorithm [35]. *k*-means algorithm achieves clustering by minimizing the within-cluster sum of squares as shown in

$$\arg\min_{s} \sum_{i=1}^{k} \sum_{x_j \in S_i} \left\| x_j - \mu_i \right\|^2, \tag{7}$$

where x_i is the video trace at index i, k is the number of sets ($k < n$, n: number of video traces), S_i is the ith set, and μ_i is the mean of S_i.

Our next step is to estimate the number of clusters or groups to consider for *k*-means clustering. PCA helps give an insight of how many clusters the video traces can be grouped into [36]. In our case, PCA suggests that we need two clusters. In order to verify the analysis results from PCA, we proceeded with computing the within-cluster sum of squares for different number of clusters. Our aim is to select the minimum number of clusters that allows the minimal possible value for the within-cluster sum of squares. By plotting these values to represent a graph similar to the previously shown *scree* test in Figure 2, the large spaces between the plotted variables and the graph *knee* values indicate the possible values are two, three, and four clusters as shown in the Figure 4(a). To further investigate the best possible number of clusters to use, we performed a hierarchical clustering to identify the number of clusters using *Ward's*



(a) Within groups sum of squares versus number of clusters

(b) Hierarchal clustering result

Figure 4: Determining number of clusters using scree test and hieratical analysis.

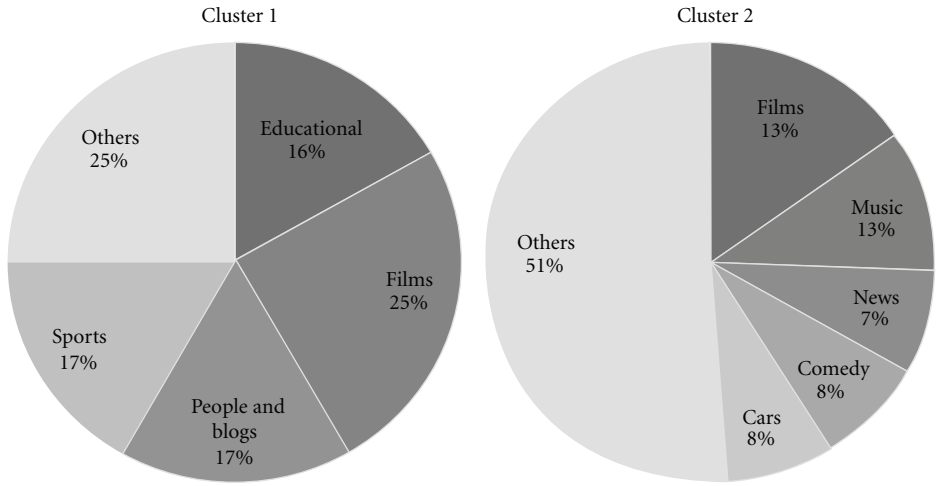


Figure 5: Distribution of movie groups over the two clusters.

method [35]. As shown in Figure 4(b), the video traces are divided into two main clusters. Our choice of grouping the video traces into two clusters was further verified by performing *silhouette* validation method [37].

By performing k-means clustering we grouped the video traces into 2 clusters. Table 5 shows the two chosen principal components corresponding to the centroids of the two clusters, and the two clusters main members. Figure 5 shows the distribution of video groups over the two clusters.

In summary, video traces that belong to cluster 2 have significantly lower mean values and have considerably low peaks compared to normal distribution, and lighter tails as indicated by their low Kurtosis values.

We also notice that films category video traces are spread across both clusters. Most blogs and sport category videos are characterized as peaky video traces because of their content. News and comedy videos are less peaky and have lower means than other movies.

To summarize, in this section, we demonstrated our results of performing both factor and cluster analysis on our collection of video traces. Both methods of analysis give us a better understanding of the distribution of the movie traces and their key statistical attributes.

4. Modeling HD Video Traces

In this section, we discuss and compare three statistical models to represent HD video traces. Several models to represent VBR (Variable Bit Rate) MPEG traffic have been proposed in the recent years. Some of the models proposed are based on Markov chain models, which are known for their inefficiency

Table 5: Clustering results using k-means clustering.

Variables	Cluster 1	Cluster 2
Mean	59,251	32,582
Kurtosis	12.028829	9.099512
No. of elements	13	39
Main video groups	Films, people and blogs, sports, educational	Films, music, news, comedy, cars

in representing the long-range dependence (LRD) characteristics of MPEG traffic [38, 39]. Due to the high influence of LRD, multiplicative processes have been considered like Fractional ARIMA (FARIMA), which have been shown to be accurate, but also computationally demanding and provide marginal improvements over ARIMA [12]. Wavelet-based prediction has been shown to require more computation resources, and to provide less accurate results than ARIMA [40]. Our aim is to select a simple to implement, accurate, and applicable model for all video traces without the need of significant statistical background.

The chosen model should not require video-specific, complex, and tedious steps. The model should be able to not only represent video frame size distribution, but also the correlation between the frames. These attributes are important to achieve the desired results and to allow the analysis of our large collection of video traces. Our pre-analysis step resulted in choosing three modeling methods: autoregressive (AR) model, autoregressive integrated moving average (ARIMA) model using the approach proposed in [13], and SAM model [14, 15]. All these models use maximum likelihood estimation to determine the model terms coefficients and consider Akaike's Information Criterion (AIC) as their optimization goal. AIC is defined as

$$\mathrm{AIC} = 2k + n\left[\ln\left(\frac{\mathrm{RSS}}{n}\right)\right], \tag{8}$$

here k is the number of parameters, n is the number of the video frames, and RSS is the residual sum of squares. AIC defines the goodness of fit for the models, considering both their accuracy and complexity defined by their number of parameters. Lower AIC values indicate better models in terms of their validity and simplicity. Each of the modeling methods is described briefly below.

4.1. AR Modeling. Autoregressive fitting takes into consideration the previous values of the fitted trace. An autoregressive model of order p can be written as

$$X_t = \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t, \tag{9}$$

where φ_i is the ith model parameter and ε_t is white noise.

We used maximum likelihood estimation (MLE) to estimate the model parameters of the AR model. Using AR to fit the video traces is a considerably simple process, but it does not always yield accurate results. Additionally, each video trace has its own set of parameters in terms of their numbers and their coefficients values.

4.2. ARIMA Modeling. Autoregressive integrated moving average model is a mathematical class model with both autoregressive and moving average terms. Moving average (MA) terms describe the correlation between the current value of the trace with the previous error terms. The integrated or differencing part of the model can be used to remove the nonstationarity of the trace.

ARIMA is usually referred as ARIMA (p, d, q) where p is the order of the autoregressive part, q is the order of the moving average part, and d is the order of differencing part. ARIMA model can be written as

$$\left(1 - \sum_{i=1}^{p} \phi_i L^i\right)(1 - L)^d X_t = \left(1 + \sum_{i=1}^{q} \theta_i L^i\right)\varepsilon_t, \tag{10}$$

where L is the lag operator and θ_i is the ith moving average parameter. We used the automatic ARIMA estimation algorithm proposed in [13], which implements a unified approach to specify the model parameters using a stepwise procedure. It also takes into consideration the seasonality of the trace to allow representing seasonal data series. This approach also results in a separate set of parameters for each video trace in terms of their numbers and their values. For the rest of this paper we will refer to this approach simply as ARIMA.

4.3. SAM Model. SAM is a mathematical model based on Seasonal ARIMA (SARIMA) models [12]. SARIMA models aim to achieve better modeling by identifying both nonseasonal and seasonal parts of data traces. SARIMA is described as

$$\mathrm{SARIMA} = (p, d, q) \times (P, D, Q)^s, \tag{11}$$

where P is the order of the seasonal autoregressive part, Q is the order of the seasonal moving average part, D is the order of seasonal differencing, and s denotes the seasonality of the time series. SAM as SARIMA model can be written as

$$\mathrm{SAM} = (1, 0, 1) \times (1, 1, 1)^s, \tag{12}$$

where s is the video trace seasonality, in our case this is equal to the frames rate.

SAM provides a unified approach to model video traces encoded with different video codec standards using different encoding settings [14, 15]. The model was developed to model mobile video traces, and we investigate in this paper its ability to model more resource-demanding HD

video traces with higher resolutions and different encoding settings. Seasonal ARIMA modeling can be represented by two main steps: defining the model order, by selecting the order of p, d, q, P, D, and Q terms, and then estimating the order coefficients using methods like maximum likelihood estimation. SARIMA models require a considerable degree of analysis and statistical background to identify the model terms order. SAM, on the other hand, has only four parameters, and therefore each model is represented with only four coefficient values, while achieving similar results to the SARIMA models calculated for each video trace [14, 15]. The values the parameters are determined using maximum likelihood estimation and optimized using *Nelder-Mead* method [41]. The four parameters are the coefficients of autoregressive, moving average, seasonal autoregressive, and seasonal moving average parts. Therefore, using SAM simplifies the analysis process that is usually required for seasonal series and removes manual processing and expert analysis requirements.

4.4. Modeling Results. After performing the modeling analysis on 54 HD video traces, we evaluated the achieved results first by simply comparing the sum of the AIC values for all the modeled video traces. We also calculated the number in which each model scored the best AIC, that is, lowest value, for a certain video trace. Additionally, we compared the three models using three statistical measures to evaluate how close the models values are to the actual traces: the mean absolute error (MAE) as shown in (13), mean absolute relative error (MARE) as shown in (14), and normalized mean square error (NMSE), as shown in (15)

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|e_i|, \tag{13}$$

$$\text{MARE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|e_i|}{x_i}, \tag{14}$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(e_i)^2}, \tag{15}$$

where N is number of frames, e_i is the modeling error at the ith frame, and x_i is the ith frame size. The results are shown in Table 6. It can be noted that SAM achieved the best results, while AR and ARIMA came in second and last place, respectively. The achieved results demonstrate two main points: SAM is superior to the other two modeling methods, and that the automated approach used with ARIMA modeling does not always yield the expected results.

Additionally, we performed several graphical comparisons for all the video traces by comparing the original video traces, their autocorrelation function (ACF) plots, and their empirical cumulative distribution function (ECDF) plots to ones achieved by the different models. Figure 6 shows an example of one of the compared video traces. As we can notice, SAM has better results and represents the traces statistical characteristics better than the other two models. For this example, modeling using AR required 12 parameters, using ARIMA required 7 parameters (two AR parameters and five MA parameters), and using SAM required only 4 parameters.

Table 6: Comparison between AR, ARIMA, and SAM using AIC.

	AR	ARIMA	SAM
Total MAE	830753	894700	641897
Total MARE	200.12	220.47	126.28
Total RMSE	1583607	1644015	1114846
Total AIC	3473929	3492401	3344490
No. of Best AIC	6	3	43

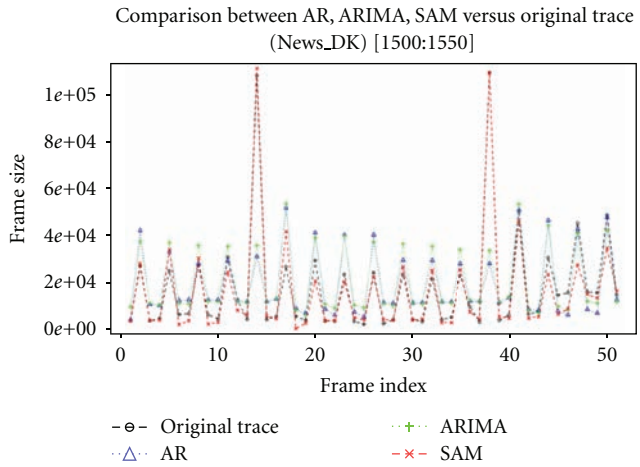
The results show that SAM has a significant advantage over the other two modeling methods especially on the seasonal transition of the video trace. This advantage is also apparent in ACF and ECDF plots comparisons. All graphical comparison results for all the HD video traces are also available through our website [20].

5. Forecasting HD Video Traffic

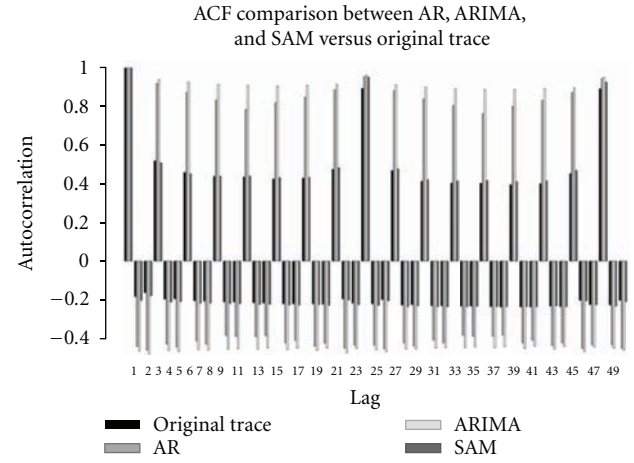
Because of the variability exhibited in video traffic and especially in AVC encoded videos, static bandwidth allocation is considered not suitable for providing high utilization of the network resources. Thus, dynamic bandwidth allocation has been considered as an alternative approach [42]. The heart of the dynamic bandwidth allocation schemes is the traffic predictor that helps in making decisions for future bandwidth allocations.

In order to evaluate the different prediction methods, we characterized different requirements for the predictor in which to operate. These requirements are set to test the abilities of these models to operate under different network traffic scenarios. The first criterion is the model's ability to correctly predict traffic to achieve long-term prediction. The prediction process itself consumes network resources. Thus, it is preferable to run the predictor as few times as possible. On the other hand, we do not need the prediction interval to be too large, because the video frame sizes change frequently and do not follow a certain pattern for a long period of time that may result in severe prediction errors. Prediction errors results in either in inefficient use of network resources, or result in an increased rate of dropped packets due to insufficient space in the receiving network buffers. We evaluated this criterion by comparing the three modeling methods using four different prediction interval lengths: 48, 72, 96, and 120 frames that translate to 2, 3, 4, and 5 seconds, respectively.

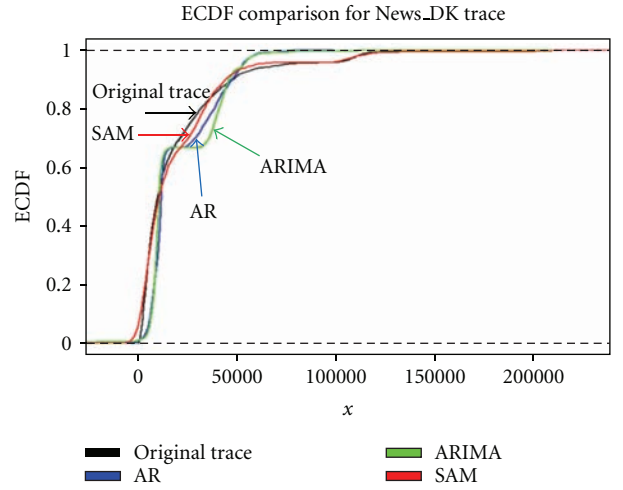
The second criterion is the ability of the predictor to capture the statistical characteristics of the movie trace by analyzing as few video frames as possible. We evaluated this criterion by comparing the prediction accuracy in the cases where the predictor has already processed 250, 500, 1000, and 1500 video frames. This translates to around 10, 20, 40, and 60 seconds, respectively.



(a) Trace comparison (frames between 1500–1600)



(b) ACF comparison (first 50 lags)



(c) ECDF comparison

Figure 6: Graphical comparisons between AR, ARIMA, and SAM.

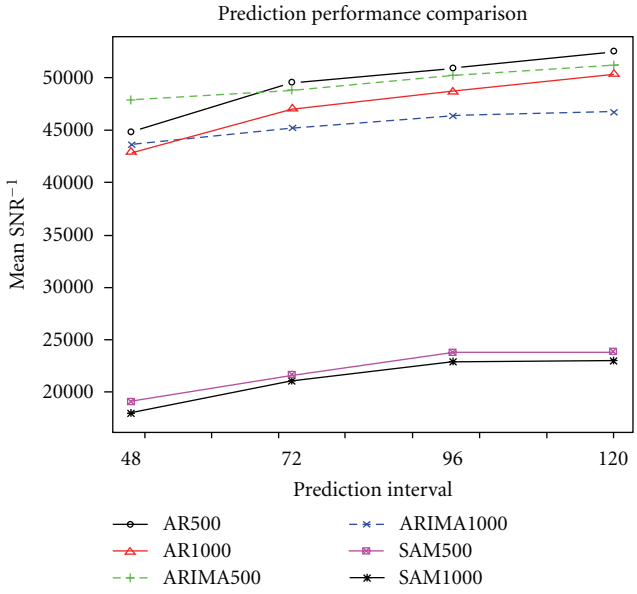


Figure 7: Comparisons between AR, ARIMA, and SAM SNR^{-1} values.

Evidently, we seek out the best predictor that can achieve the best prediction accuracy for the longest prediction window with the least number of frames to be analyzed. We chose the commonly used noise to signal (SNR^{-1}) ratio as our prediction accuracy metric. SNR^{-1} computes the ratio between the sum of squares of the prediction errors, and the sum of squares of the video frame size. SNR^{-1} can be depicted as

$$\mathrm{SNR}^{-1} = \frac{\sum (e)^2}{\sum (\mathrm{size})^2}, \tag{16}$$

where e is the prediction error, and *size* is the video frame size.

Figure 7 shows a summary of the main results. As seen in this figure, the prediction error is directly related to the increase of the prediction window size. It also shows that the increase of the predictor knowledge, as represented in the number of frames processed, provides better prediction accuracy. It is obvious from the figure that SAM provides significant improvements over the other two methods. Table 7 shows the improvements SAM provides over AR and ARIMA when 1000 frames are processed. SAM improves up to 55% over AR, and 53.3% over ARIMA.

To better understand the reasons behind the observed improvement, we plot the three models predictions for a prediction window of 48 after processing 1000 of video frames. As shown in Figure 8, SAM not only manages to predict the video frames accurately, it is the only one that can predict the significant transitions of the frame sizes. SAM can also provide accurate results with relatively fewer numbers of frames. For instance, SAM results with 1500 preprocessed frames have only 4.7% improvement over SAM with 250 preprocessed frames [19].

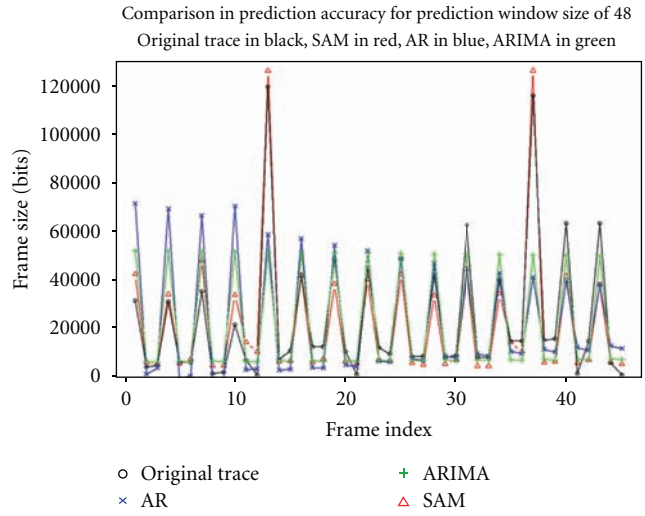


Figure 8: Prediction comparison between AR, ARIMA, and SAM.

Table 7: SNR^{-1} comparison between AR, ARIMA, and SAM.

	AR 1000	ARIMA1000	SAM 1000
SNR^{-1} (avg)	47180	45457	**21220**
Improvement over AR	—	3.6%	**55**%
Improvement over ARIMA	−3.6%	—	**53.3**%

We further investigated the possibility of using SAM with even fewer numbers of frames. Theoretically, SAM needs a minimum of 29 frames as suggested in [43]. Our results showed that we need at least 100 frames to achieve the desired accuracy. With SAM, using 1500 frames provided only 1% improvement over using 100 frames on average. Thus, based on our results, we recommend using SAM with 100 frames (~4 seconds) to predict the subsequent 120 frames (5 seconds). This means that a dynamic bandwidth allocation scheme needs only to negotiate the allocation once every 5 seconds.

In this section we compared three possible models that can be used to achieve the desired prediction accuracy with HD video traces. Our results showed that SAM has a clear edge over the two other models. In the next section we discuss some of the developed tools.

6. SAM-Based Developed Tools

In this section we demonstrate the design and implementation of two SAM-based tools: SAM-based traffic generator that can be used to generate HD video traces for video streaming simulations, and GUI interface to facilitate the analysis of video traces.

6.1. SAM-Based Trace Generator. As we have mentioned before, SAM allows researchers to represent the video traces using only four parameters. In addition to that we need a fifth

parameter to help initialize the simulation process needed for the traffic generation. The fifth parameter is the standard deviation of the modeling error.

In *R* [44], there are two functions that can be used to generate time-series points based on ARIMA models: *arima.sim* and *garsim* included in the *gsarima* package [45]. Unfortunately, these two functions can only simulate ARIMA models and not work with SARIMA (seasonal ARIMA) models. To overcome this obstacle, we converted SARIMA model to an infinite series of AR coefficients [12, 46]. The *gsarima* package provides a function "*arrep*" that is capable of such conversion. From our experience, we found that 250 AR coefficients are sufficient to provide good results. We implemented the SAM-based generator using C#. The generator is based on the equation developed in [47].

Figure 9 shows a CDF comparison between the trace obtained from our trace generator and the actual trace. The provided trace generator implementation is also available for the research community to improve and adjust to different simulation setups.

6.2. SAM-Based Video Trace Analyzer. To ease the analysis of video traces and the comparison of SAM model against the original trace, we developed a simple GUI, shown in Figure 10, that allows the users to load the video trace frame size information from a text file. SAM analyzer then processes the information and calculates the seasonality of the trace, its SAM parameters, and its AIC value.

The user can plot the ACF and PACF graphs of the video trace. In addition, the user can plot original video trace versus SAM generated trace comparison graphs for ACF and ECDF. Figure 11 shows an example an example of the comparison graphs generated by SAM trace analyzer. Additional comparisons can be added upon user needs. SAM trace analyzer is implemented using C#. Our implementation provides an interface to *R* compiled code to allow full utilization of its capabilities.

In this section we illustrated the usage of two of our developed tools. In the next section we discuss the importance of our contribution and conclude the paper.

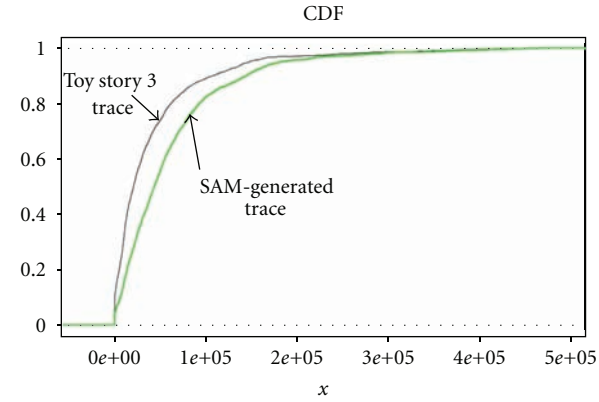


Figure 9: CDF comparison between SAM-generator and actual trace.

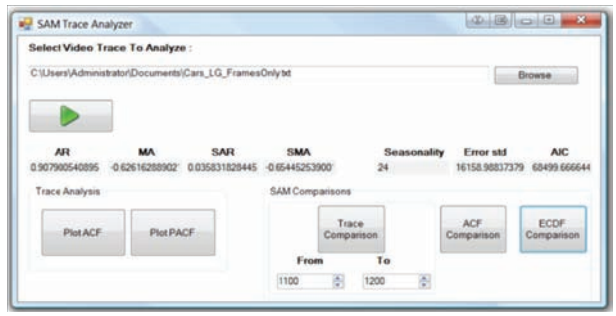


Figure 10: SAM-based video trace analyzer GUI.

7. Conclusions

In this paper, we presented our work of encoding, analyzing, and modeling over 50 HD video traces that represent a wide spectrum of statistical characteristics.

We can summarize the key contributions of this paper in the following points.

(1) We collected over 50 HD video traces from YouTube website that represents a wide variety of video traces. We encoded these traces using AVC standard with the most common settings supported by experts' recommendations. These traces provide the research community with the means to test and research new methods to optimize network resources, and especially using dynamic bandwidth allocation. All the video traces and the developed tools are available to the research community through our website [20].

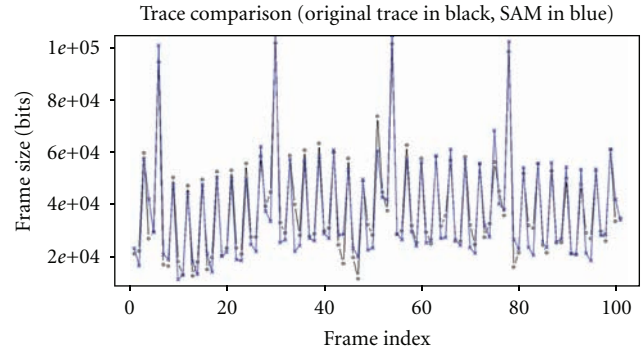
(2) We performed a full statistical analysis to show the variance of the collected video traces using the most common quantitative measures.

(3) We performed a factor analysis to determine the principal components that define a HD video trace. We concluded that both Mean and Kurtosis values can be considered as the two main principal components. Our analysis has shown that both Kurtosis and Skewness values are important in defining a HD video trace, contrary to what has been considered before for MPEG1 encoded videos.

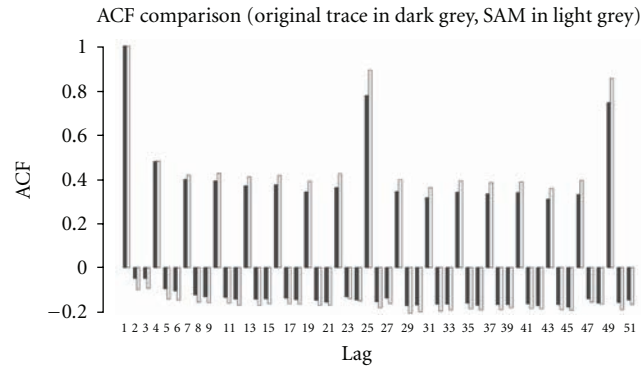
(4) We performed a cluster analysis on our collection of HD videos using *k*-means clustering. Our results showed that we can group these movies into two main clusters. We supported our results using different graphical and nongraphical methods.

(5) We compared three modeling methods in their ability to model our collection of HD video traces. Our results showed that SAM has a clear advantage over both AR and ARIMA models in both accuracy and simplicity as represented in its AIC values.

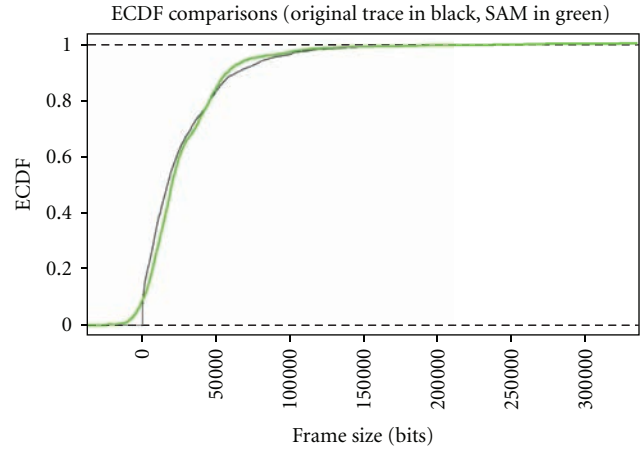
(6) We have also compared these methods in their ability to forecast video traffic. Our prediction analysis was based on several factors to ensure that the chosen model is capable of providing the best results under the lowest requirements. Our results showed once



(a) Trace Comparison (frames between 1100–1200)



(b) ACF Comparison (first 50 lags)



(c) ECDF Comparison

Figure 11: An example of SAM trace analyzer generated comparison plots.

again that SAM has a significant improvement over both AR and ARIMA, where it provided at least 50% better SNR^{-1} values.

(7) Finally we illustrated the implementation of two of our developed tools. We showed the ability of the SAM-based generator of generating HD video traces that can be configured and used in different simulation scenarios.

This contribution provides the initial steps to achieve a better dynamic bandwidth allocation schemes designed to optimize bandwidth utilization with the presence of the high-demanding HD video streams.

References

[1] Google, "YouTube HD video section," June 2010, http://www.youtube.com/HD.

[2] C. Albrecht, "Survey: Online Video Up to 27% of Internet Traffic," October 2009, http://tinyurl.com/yzpzoew.

[3] Cisco, "Cisco VNI: Forecast and Methodology, 2010–2015," February 2012, http://tinyurl.com/3p7v28.

[4] Comscore Press Release, http://tinyurl.com/l4o3rs.

[5] Comscore Press Release in November 2009, http://news.websitegear.com/view/149267.
[6] Hulu, "Hulu Website," June 2011, http://www.hulu.com.
[7] Netflix, "DVD Rental and HD video streaming service," June 2011, http://www.netflix.com.
[8] A. Adas, "Supporting real time VBR video using dynamic reservation based on linear prediction," in *Proceedings of 15th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '96)*, pp. 1476–1483, March 1996.
[9] M. Wu, R. A. Joyce, H. S. Wong, L. Guan, and S. Y. Kung, "Dynamic resource allocation via video content and short-term traffic statistics," *IEEE Transactions on Multimedia*, vol. 3, no. 2, pp. 186–199, 2001.
[10] Y. Liang and M. Han, "Dynamic bandwidth allocation based on online traffic prediction for real-time MPEG-4 video streams," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, Article ID 87136, 2007.
[11] G. van der Auwera, P. T. David, and M. Reisslein, "Traffic characteristics of H.264/AVC variable bit rate video," *IEEE Communications Magazine*, vol. 46, no. 11, pp. 164–174, 2008.
[12] C. Chatfield, *The Analysis of Time Series: An Introduction*, Chapman & Hall/CRC, 6th edition, 2003.
[13] R. J. Hyndman and Y. Khandakar, "Automatic time series forecasting: the forecast package for R," *Journal of Statistical Software*, vol. 27, no. 3, pp. 1–22, 2008.
[14] A. K. Al Tamimi, R. Jain, and C. So-In, "Modeling and generation of AVC and SVC-TS mobile video traces for broadband access networks," in *Proceedings of the 1st Annual ACM SIGMM Conference on Multimedia Systems (MMSys '10)*, pp. 89–98, February 2010.
[15] A. Al Tamimi, C. So-In, R. Jain et al., "Modeling and resource allocation for mobile video over WiMAX broadband wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 3, pp. 354–365, 2010.
[16] P. Manzoni, P. Cremonesi, and G. Serazzi, "Workload models of VBR video traffic and their use in resource allocation policies," *IEEE/ACM Transactions on Networking*, vol. 7, no. 3, pp. 387–397, 1999.
[17] O. Rose, "Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems," in *Proceedings of the 20th Conference on Local Computer Networks*, no. 16–19, pp. 397–406, October 1995.
[18] L. L. Laetitia, *MPEG-4 AVC traffic analysis and bandwidth prediction for broadband cable networks*, M.S. thesis, Georgia Tech, 2008.
[19] A. K. Al Tamimi, R. Jain, and C. So-In, "Modeling and prediction of high definition video traffic: a real-world case study," in *Proceedings of the 2nd International Conferences on Advances in Multimedia (MMEDIA '10)*, pp. 168–173, Athens, Ga, USA, 2010.
[20] A. Al-Tamimi and R. Jain, "SAM model Traces website," June 2011, http://www.cse.wustl.edu/~jain/sam/index.html.
[21] K. Jack, *Video Demystified*, HighText, 2nd edition, 1996.
[22] G. E. P. Box and G. Jenkins, *Time Series Analysis: Forecasting and Control*, Holden-Day, 1976.
[23] The Top 5 video streaming websites, March 2010, http://www.techsupportalert.com/top-5-video-streaming-websites.htm.
[24] MediaInfo, "MediaInfo supplies technical and tag information about your video or audio files," June 2011, http://mediainfo.sourceforge.net/en.
[25] J. Ozer, "Producing H.264 Video for Flash: An Overview," March 2010, http://www.streaminglearningcenter.com/articles/producing-h264-video-for-flash-an-overview.html.
[26] Digital Rapids, April 2012, http://dr6.sitesystems.ca/downloads/docs/DR_Studio_AVC.pdf.
[27] FFMPEG Coding Library, Cross-platform solution to record, convert and stream audio and video, http://ffmpeg.org.
[28] x264 Encoder, March 2010, http://www.videolan.org/developers/x264.html.
[29] JM Reference Software, March 2010, http://iphome.hhi.de/suehring/tml/.
[30] B. S. Everitt, *An R and S-Plus® Companion to Multivariate Analysis*, Springer, 2007.
[31] H. T. Kaiser, "The application of electronic computers to factor analysis," *Educational and Psychological Measurement*, vol. 20, pp. 141–151, 1960.
[32] R.B. Cattel, "The scree test for the number of factors," *Multivariate Behavioral Research*, vol. 1, no. 2, pp. 245–276, 1966.
[33] G. Raîche, M. Riopel, and J. Blais, "Non graphical solutions for the cattell's scree test," in *Proceedings of the Annual Meeting of the Psychometric Society*, Montreal, Canada.
[34] G. Kootstra, "Project on exploratory Factor Analysis applied to foreign language learning," 2004.
[35] M. Norusis, *SPSS 17.0 Statistical Procedures Companion*, Prentice Hall, 2009.
[36] C. Ding and X. He, "K-means clustering via principal component analysis," in *Proceedings of the 21t International Conference on Machine Learning (ICML '04)*, vol. 69, 2004.
[37] Cluster Validity Algorithms, October 2009, http://tinyurl.com/yj8jz9w.
[38] A. M. Dawood and M. Ghanbari, "Content-based MPEG video traffic modeling," *IEEE Transactions on Multimedia*, vol. 1, no. 1, pp. 77–87, 1999.
[39] Y. Sun and J. N. Daigle, "A source model of video traffic based on full-length VBR MPEG4 video traces," in *Proceedings of IEEE Global Telecommunications Conference*, vol. 2, p. 5, 2005.
[40] H. Feng and Y. Shu, "Study on network traffic prediction techniques," in *Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing (WCNM '05)*, pp. 1041–1044, September 2005.
[41] J. A. Nelder and R. Mead, "A simplex algorithm for function minimization," *Computer Journal*, vol. 7, pp. 308–313, 1965.
[42] H. Zhao, N. Ansari, and Y. Shi, "Efficient predictive bandwidth allocation for real time videos," *IEICE Transactions on Communications*, vol. 86, no. 1, 2003.
[43] R. Hyndman and A. Kostenko, "Minimum sample size requitrements for seasonal forecasting models," *Foresight*, vol. 6, pp. 12–15, 2007.
[44] The project R of statstical computing, June 2011, http://www.r-project.org.
[45] O. Briet, "Gsarima: Two functions for Generalized SARIMA time series simulation," June 2011, http://cran.fyxm.net/web/packages/gsarima/index.html.
[46] D. Montgomery, *Forecasting and Time Series Analysis*, McGraw-Hill, 1990.
[47] A. K. Al-Tamimi, R. Jain, and C. So-In, "Dynamic resource allocation based on online traffic prediction for video streams," in *Proceedings of IEEE 4th International Conference on Internet Multimedia Services Architecture and Application (IMSAA '10)*, Bangalore, India, December 2010.