2010 Second International Conferences on Advances in Multimedia

Modeling and Prediction of High Definition Video Traffic: A Real-World Case Study

Abdel Karim Al Tamimi
Department of Computer Science and Engineering
Washington University in St. Louis
Saint Louis. MO USA
aa7@cse.wustl.edu

Raj Jain
Department of Computer Science and Engineering
Washington University in St. Louis
Saint Louis. MO USA
jain@cse.wustl.edu

Chakchai So-In
Department of Computer Science and Engineering
Washington University in St. Louis
Saint Louis. MO USA
cs5@cse.wustl.edu

Abstract— **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 comparing over 50 HD video traces. We show through our results that our model: simplified seasonal ARIMA (SAM) provides a good representation for HD videos, and it provides significant improvements in prediction accuracy over other regressions methods. In addition, we discuss our methodology to collect and encode our library of HD video traces. We describe the tools that we have created and used in generating create and analyzing these traces. We have made these tools, along with our large collection of HD video traces, available for the research community. We illustrate the simplicity of our approach and we discuss the importance of our modeling and prediction method and its impact on other areas of study.**

Keywords- Modeling Video Traces, Traffic and Workload Modeling and Characterization, High Definition Video Traces. AVC Video Encoding, Seasonal ARIMA model, SAM Model, Video Traffic Prediction.

I. INTRODUCTION

Network traffic analysis is an important step in any communications system design. Network researchers must use a valid traffic load to analyze a system or a network setup correctly. There are two sources of traffic loads: sample traces that are extracted from real traffic, and synthetic workloads generated from mathematical models that imitate the traffic's statistical behavior to produce the correct load. The two approaches have their individual merits. Trace- driven simulations are considered credible as they represent an actual traffic load. But they are static and provide only a point representation of the workload space. Thus, several traces are needed to represent the potential workloads correctly under different conditions, adding complexity to the approach [4].

Another important point is the ability to run the simulation process for a longer or shorter duration than the trace length. In short simulations, if a portion of the selected trace is used, the researchers need to make sure that the chosen segment is representative of the trace, which is a challenging task, especially in highly time-variant traces like video traces. In long simulations, the required trace length is typically multiple times longer than the available traces. In this case, the researchers need to find a way to loop through the trace to provide the necessary trace length. Moreover, as we discuss in this paper, obtaining video traces is a resource-consuming process. Alternatively, model-based trace generators can be easily modified and adapted to different simulation settings, provided that the algorithm used to model and generate video traces is valid. Several attempts have been made in the recent past to address this topic [5-7]. The previous approaches have produced models that apply only to a movie or a specific movie scene. In addition, some of the models have many parameters that need to be adjusted to represent different video characteristics. For example, in [8] eleven parameters are needed to model a single movie.

We have developed a simple and accurate model to represent a wide variety of movie traces. The simplified seasonal ARIMA model, or SAM, is based on the seasonal autoregressive integrated moving average (ARIMA) model [2, 21, 23]. The model is unique in four respects: first, it is simple in that it requires only five parameters to represent a movie trace. Second, it is general in that the same model applies to numerous movies and videos. Third, it can be used to predict future video traffic to aid in dynamic bandwidth allocation solutions. Fourth, it can be used to generate traces that closely resemble the modeled traces.

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 our results that are based over 50 HD video traces from the popular video hosting website YouTube [3].

One of the main challenges in developing a valid video workload model is to find an adequate number of traces to test the 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 [1].

In addition to analyzing, and modeling these video traces,

978-0-7695-4068-9/10 $26.00 © 2010 IEEE
DOI 10.1109/MMEDIA.2010.10

we provide a trace generator based on our model that can be used to generate user-defined traces with the desired statistical characteristics. The trace generator can also be used to produce a new movie trace that represents a blend of different video characteristics. Figure 1 shows the steps we took in analyzing and modeling the selected videos with their corresponding outputs.

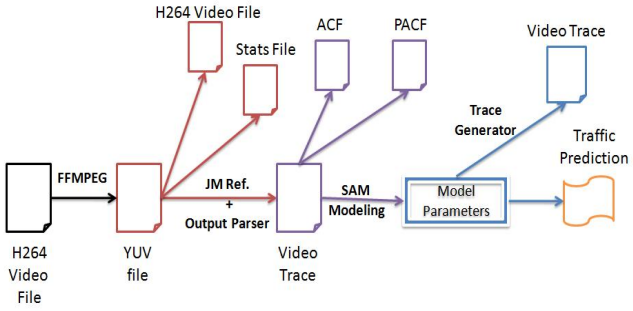


Figure 1. Modeling, analyzing, and generating video traces processes

The process starts with a YouTube video, which is then converted to a YUV raw video. The raw video is subsequently encoded with AVC to produce an encoded movie file, its encoding statistics file, and a full verbose description of the encoding process. The verbose output is then parsed to get the video frames information, which is then modeled using SAM. The video trace is used also to produce the video's 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 plot for a lag k is the plot of the autocorrelation between X_t and X_{t-k} that is not accounted for by lag 1 to k-*1* inclusive.

The obtained SAM parameters for each video can be used to either predict future traffic, or to produce a movie trace. 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 II provides a simple introduction to SAM and its parameters. Section III illustrates our approach in selecting and encoding our collection of video files. Section IV describes the results we obtained using our modeling approach. Section V compares different approaches to achieve an accurate video traffic prediction. Section VI illustrates our trace generator design and its implementation. Finally, we conclude the paper and give some insight to the impact of our presented results.

II. Overview of SAM Model

We have proposed a statistical model to represent mobile video traffic accurately, called SAM [2, 21, 23]. SAM is based on seasonal ARIMA models. ARIMA models are used in time series analysis, and they have three main parts: an autoregressive part (AR), an integrated or differencing part (I), and a moving average part (MA). ARIMA models are usually represented as ARIMA (p, d, q), where p is the order of the autoregressive part, d is the order of the differencing part, and q is the order of the moving average part. ARIMA models can be implemented using simple equations. For example, ARIMA (1, 1, 1) can be described as

$$y(t) = w(t) + y(t-1) + \phi(y(t-1) - y(t-2)) - \theta w(t-1) \quad (1)$$

where $w(t)$ is the error term at time t, ϕ is the coefficient of AR, and θ is the coefficient of MA of the ARIMA model. Seasonal ARIMA (SARIMA) is an extension of the ARIMA model to express series that exhibit periodic or seasonal behavior. SARIMA is described as

$$SARIMA = (p,d,q)\times(P,D,Q)^s \quad (2)$$

where P, D, and Q represent the order of the seasonal AR (SAR) part, the seasonal differencing part, and the order of the seasonal MA (SMA) part, respectively. S represents the seasonality of the series (e.g., month seasonality in a year is 12) [10, 11]. After extensive analysis of video traces encoded with MPEG4-Part2 coding for mobile devices, we determined that the following simplified seasonal ARIMA model provides a good representation of such videos

$$SAM = ARIMA(1,0,1)\times(1,1,1)^z \quad (3)$$

where z is the seasonality of the video trace and it is equal to group of pictures (GoP) size in MPEG4-Part2 [2, 23]. We extended our research to include both AVC and SVC-TS encoded videos [21]. One of key characteristics of this model is that only five parameters are used to represent any video trace. These parameters are: AR coefficient, MA coefficient, SAR coefficient, SMA coefficient, and the standard deviation of the error terms used to produce the initial vector for the Monte Carlo simulation in the trace generation process.

SAM provides a unified approach to model video traces encoded with different video codec standards using different encoding settings [21, 23]. SARIMA models require multipart analysis to identify the model parameters number and their values. SAM, on the other hand, has a unified and automated approach to model video traces.

Despite that SAM originally was proposed for mobile video traces, we show in this paper how it can also be applied to various HD video traces with higher resolutions and more demanding encoding settings. We will discuss in the next section our approach to select and encode our collection of YouTube HD videos.

III. 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 sharing site on the Internet. 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 some of the most visited videos in YouTube HD section [3]. 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* [13] 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 [14, 15] 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. We performed the converting process using the open source coding library FFMPEG [12].

To convert YUV files to the H.264/AVC format, we tested two publically available encoding libraries: x264 [19] and JM reference software [20]. Though x264 is significantly faster than JM reference software, it provided us with less information about the encoding process. Table I lists some of the chosen encoding parameters using JM reference software.

TABLE I
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

As mentioned before, these parameters were chosen to represent the majority of the videos we have collected. We used in our encoding Instantaneous Decoding Refresh (IDR) frames. IDR frames are special type of I frames that allow better seeking precision and thus enhance the user's experience. In our encoding process, we used *closed-GOP* setting [15] 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. 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 (QP) used are the default values for the encoder.

The parameter *NumberReferenceFrames* sets the maximum number of reference frames stored in the decoder buffer. 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-consuming and resource-consuming process. The encoding of a single video file took on average 37 hours, with an average encoding rate of 0.02fps. The average size of a raw (YUV 4:2:0) video file is around 4GB. These figures support our conclusion 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 information we need for the next steps of our analysis and modeling. In the next section, we discuss some of our results of modeling HD video traces using SAM.

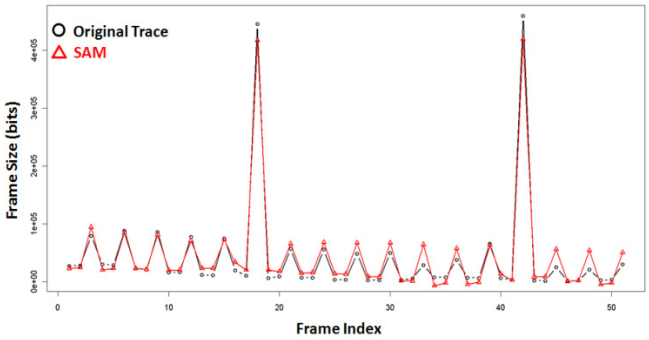
IV. MODELING HD VIDEOS TRACES WITH SAM

The inputs at this stage are the outputs of the parsing tool from the previous stage. Our video collection represents a wide spectrum of video statistical properties. Table II shows few of the statistical characteristics for some of these videos. As shown in the table, the selected videos show diversity in their statistical characteristics. The *hurst exponent* indicates the trace ability to regress to the mean, with higher values indicating a smoother trend, less volatility, and less roughness. Its value varies between 0 and 1.

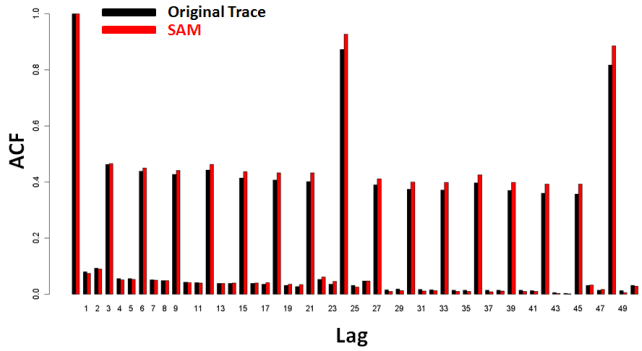
TABLE II
STATISTICAL CHARACTERISTICS COMPARISON FOR SOME VIDEO TRACES

Parameter/Movie	Mega Man Project	Relaxation Hawaii Beaches (2-4)	30 DC Day 07	Tiger Woods 70th Career Title
Mean (bits)	87024	678872	12020	48837
Min	288	296	296	288
Max	803736	678872	296560	420552
Median	14148	50196	448	20112
Std	163666	93421	47915	68451
Hurst Exponent	0.90726	0.90283	0.49893	0.82586
No. of Frames	3634	9388	3016	2260
Video Category	Gaming	Music	Educational	Sports

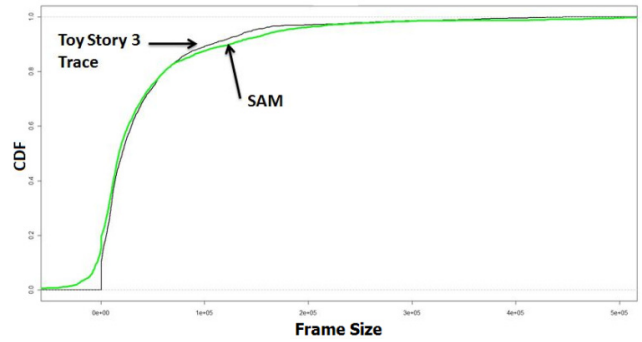
Our results showed that SAM is capable of modeling our collection of HD video traces. As an example, we illustrate the capability of the SAM model to represent one of the video traces. Figure 2 shows the results of modeling *Toy Story 3* movie trailer using different graph comparisons. Figure 2(a) shows both the actual video trace and that generated by the model. Figure 2 (b) and (c) show that the ACF and the cumulative distribution function (CDF) graphs comparisons are quite close. Similar graphs for the other video traces help us conclude that SAM is a valid model and is capable of representing the modeled video traces.



(a) Actual trace comparison



(b) ACF comparison



(c) CDF comparison

Figure 2. Comparisons between the original trace and SAM model results

Due to the space limit of this paper, we could only share a subset of the results here. The reader is invited to view the rest of the results by visiting our website [1]. In the next section we illustrate the ability of SAM to predict HD video traffic under different scenarios.

V. VIDEO TRAFFIC PREDICTION

Because of the variability exhibited in video traffic and especially in AVC encoded videos, static bandwidth allocation is considered not suitable to optimize the utilization of the network resources. Thus, dynamic bandwidth allocation has been considered as an alternative approach [27]. The heart of the dynamic bandwidth allocation schemes is a traffic predictor that helps in making decisions for future bandwidth allocations.

As we have shown in the previous section, SAM can accurately model the video traces encoded with AVC standard targeted for more demanding application. In this section, we compare the ability of SAM to predict video traffic versus two regression methods: autoregressive (AR) and optimized autoregressive integrated moving average (ARIMA). First, we start by giving a brief introduction about the two compared regression methods against SAM.

A. AR Model with Maximum Likelihood Estimation

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

$$X_t = \sum_{i=1}^{p} \phi_i X_{t-i} + \varepsilon_t \tag{4}$$

where φ_i is the i-th model parameter, and ε_t is white noise. We use maximum likelihood estimation (MLE) to estimate the model parameters of the AR model. Using AR to fit the video traces is considerably simple process, but it does not always yield accurate results. Additionally, each video traces has its own set of parameters in terms of their numbers and their values.

B. ARIMA Model with Unified Approach

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 non-stationarity of the trace. As mentioned before, ARIMA is usually referred as ARIMA(p,d,q), and ARIMA model can be written as:

$$\left(1-\sum_{i=1}^{p}\phi_i L^i\right)(1-L)^d X_t = \left(1+\sum_{i=1}^{q}\theta_i L^i\right)\varepsilon_t \tag{5}$$

where L is the lag operator, and θ_i is the i-th moving average parameter. We used *auto.arima* statistical method from *forecast* package [25], which implements a unified approach to specify the model parameters. This approach takes into consideration the seasonality of the video trace. This method also results in a separate set of parameters for each video trace in terms of their numbers and their values. For the rest of the paper we will refer to this approach simply as ARIMA. All the compared models: AR, ARIMA and SAM use akaike's information criterion (AIC) as the model's optimization criterion. AIC takes into consideration both the model accuracy and the model complexity as represented by its number of parameters. AIC can be described as:

$$AIC = 2k + n\left[\ln\left(RSS/n\right)\right] \tag{6}$$

where k is the number of parameters, n is the number of the video frames, and RSS is the residual sum of squares.

In order to evaluate the different prediction methods, we characterize different requirements for the predictor in which to operate. These requirements are set to test the abilities of these methods to operate under different network configurations. The first criterion is the method'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 window to be too large, because the video frame sizes changes frequently and do not follow a certain pattern for a long period that may results in sever prediction errors. We evaluate this criterion by comparing the three methods using four different prediction window lengths: 48, 72, 96, and 120 frames, that translates 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 evaluate 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 into 10, 20, 40, and 60 seconds respectively.

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 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:

$$SNR^{-1} = \frac{\sum (e)^2}{\sum (size)^2} \qquad (7)$$

where e is the prediction error, and $size$ is the video frame size. Figure 3 shows a summary of the main results. As figure 3 shows, 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 III shows an example of the improvements SAM provides over AR, and ARIMA. SAM improves up to 55% over AR, and 53.3% over ARIMA.

TABLE III
SNR^{-1} COMPARISON BETWEEN AR, ARIMA, AND SAM

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

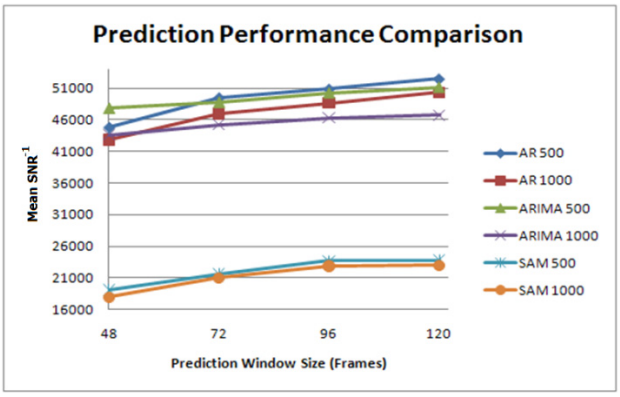


Figure 3. Comparisons between AR, ARIMA, and SAM SNR^{-1} values

To better understand the reasons behind the noted improvement, we plot the three models predictions for a prediction window of 48 after processing 1000 video frames. As shown in figure 4, 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.

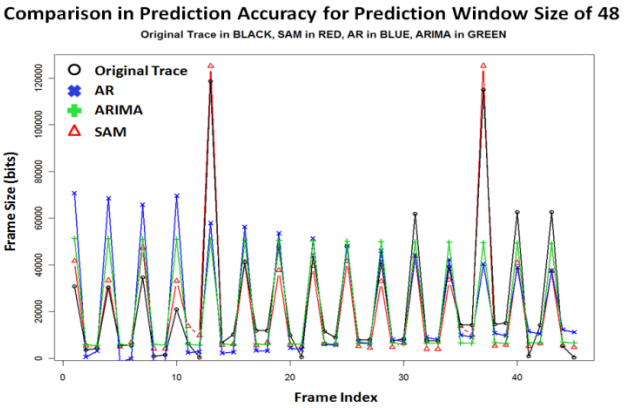


Figure 4. Prediction comparison between AR, ARIMA, and SAM

We further investigate the possibility of using SAM with even fewer numbers of frames. Theoretically, SAM needs a minimum of 29 frames as suggested in [26]. We discovered that we need at least 100 frames to achieve the desired results. With SAM, using 1500 frames provided only 1% improvement over using 100 frames on average. Thus, we recommend using SAM with at least 100 frames (~4 seconds) to predict the subsequent 120 frames (5 seconds).

VI. SAM BASED TRACE GENERATOR

As we have mentioned before, SAM allows researchers to represent the video traces using only five parameters. In R [16], there are two functions that can be used to generate time-series points based on ARIMA models: *arima.sim* and *garsim* through the *gsarima* package [17]. Unfortunately, these two functions can only simulate ARIMA models and

not SARIMA (seasonal ARIMA) models. To overcome this obstacle, we converted SARIMA model to an infinite series of AR coefficients. 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. This approach helps to simplify model simulations. For more information, the readers can refer to [10, 18]. We implemented the SAM-based generator using C# [21].

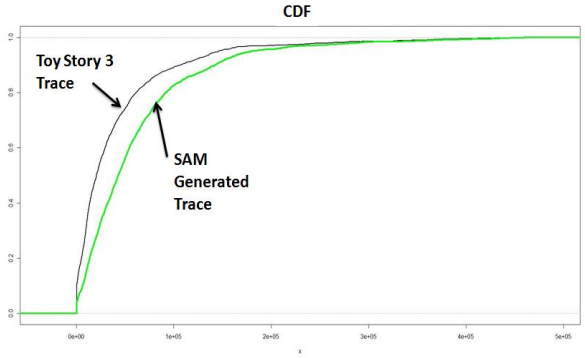


Figure 5. CDF comparison between SAM-generator and actual trace

Figure 5 shows a CDF comparison between the trace obtained from our trace generator and the actual trace. The provided trace generator implementation is available for the research community to improve and adjust to different simulation setups. In the next section we discuss the importance of our contribution and conclude the paper.

VII. CONCLUSIONS

In this paper, we presented our work of analyzing, encoding and modeling over 50 HD video traces that represent a wide spectrum of statistical characteristics. We discussed our methodology to collect and encode our HD video library. We also presented our results in modeling these traces using our model. We showed that SAM is capable of modeling different video traces that have diverse statistical characteristics.

We compared the prediction ability of SAM against two of the most common regression methods: AR and ARIMA under various prediction settings. We showed how SAM provides a significant improvement over these two methods. Additionally, we performed further analysis to determine the minimal requirements for SAM to achieve accurate results. Such predictor is essential for dynamic bandwidth allocation schemes to allow better network resources utilization. We showed the results of using our SAM-based trace generator. Our results provide the research community with the means to test and research new methods to optimize network resources.

REFERENCES

[1] SAM model Traces website, URL= http://www.cse.wustl.edu/~jain/sam/index.html.

[2] A. Al Tamimi, R. Jain, C. So-In, "SAM: A Simplified Seasonal ARIMA Model for Mobile Video over Wireless Broadband Networks," IEEE ISM 2008, pp.178-183, 2008.

[3] YouTube HD video section, URL=http://www.youtube.com/HD.

[4] R. Jain, "The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling", Wiley 1991.

[5] A.M. Dawood, and M. Ghanbari, "Content-Based MPEG Video Traffic Modeling," IEEE Transactions on Multimedia, Volume 1, Issue 1, Mar 1999, Page(s):77–87.

[6] Y. Sun, and J.N. Daigle, "A Source Model of Video Traffic Based on Full-Length VBR MPEG4 Video Traces," GLOBECOM'05, Volume 2, 28 Nov.-2 Dec. 2005, 5 pp.

[7] O. Lazaro, D. Girma, and J. Dunlop, "H.263 Video Traffic Modeling for Low Bit Rate Wireless Communication," IEEE PIMRC 2004. Volume 3, Issue, 5-8. pp(s): 2124 – 2128.

[8] X. Huang, Y. Zhou and R. Zhang, "A Multiscale Model for MPEG-4 Varied Bit Rate Video Traffic," IEEE transactions on broadcasting. Volume:50, Issue: 3, page(s): 323- 334.

[9] Mobile Devices: Video Traces, Arizona State University. URL= http://trace.eas.asu.edu/.

[10] D. Montgomery,"Forecasting and time series analysis," McGraw-Hill 1990.

[11] G. Box, G. M. Jenkins, and G. C. Reinsel, "Time series analysis : forecasting and control," J.Wiley, 2008.

[12] FFMPEG Coding Library, Cross-platform solution to record, convert and stream audio and video. URL=http://ffmpeg.org.

[13] MediaInfo, MediaInfo supplies technical and tag information about your video or audio files, URL=http://mediainfo.sourceforge.net/en.

[14] J. Ozer, "Producing H.264 Video for Flash: An Overview," URL= http://www.streaminglearningcenter.com/articles/producing-h264-video-for-flash-an-overview.html.

[15] Digital Rapids, URL= http://www.digital-rapids.com/downloads/docs/DR_AVC_Flash9.pdf.

[16] The project R of statstical computing, URL=http://www.r-project.org/

[17] Gsarima: Two functions for Generalized SARIMA time series simulation, http://cran.fyxm.net/web/packages/gsarima/index.html

[18] C. Chatfield, "The Analysis of Time Series: An Introduction," Sixth Edition Chapman & Hall/CRC 2003

[19] x264 Encoder, URL= http://www.videolan.org/developers/x264.html

[20] JM Reference Software, URL=http://iphome.hhi.de/suehring/tml/

[21] A. Al Tamimi, R. Jain, C. So-In, " Modeling and Generation of AVC and SVC-TS Mobile Video Traces for Broadband Access Networks," ACM Multimedia Systems 2010.

[22] G. E. P. Box, G. Jenkins ,"Time Series Analysis: Forecasting and Control," Holden-Day 1976.

[23] A. Al Tamimi, C. So-In, R. Jain, "Modeling and Resource Allocation for Mobile Video over WiMAX Broadband Wireless Networks," IEEE JSAC, Special issue on Wireless Video Trans., Vol.28, NO. 3, April 2010.

[24] A. C. Harvey, C. R. McKenzie, "Algorithm AS182. An algorithm for finite sample prediction from ARIMA processes". *Applied Statistics* 31, 180–187.

[25] R. Hyndman, Y. Khandakar, "Automatic Time Series Forecasting: The forecast Package for R," Journal of Statistical Software, Vol. 27, Issue 3, Jul 2008.

[26] R. Hyndman, A. Kostenko, "Minimum Sample Size Requitrements for Seasonal Forecasting Models", Foresight: the International Journal of Applied Forecasting (2007), 6, 12-15.

[27] H. Zhao, N. Ansari, Y. Shi, "Efficient Predictive Bandwidth Allocation for Real Time Videos", IEICE TRANS. COMMUN., vol. E86-B, No.1 2003.