Workload Models of VBR Video Traffic and Their Use in Resource Allocation Policies

Pietro Manzoni, *Member, IEEE*, Paolo Cremonesi, and Giuseppe Serazzi, *Member, IEEE*

***Abstract*—The load generated by new types of communications services related to multimedia and video transmission is becoming one of the major sources of traffic in WAN networks. Modeling this type of load is a prerequisite for any performance study. In this paper, we approach the load-characterization problem from a global point of view by analyzing a set of 20 video streams. We developed resource-, subject-, and scene-oriented characterizations of coded video streams. We have also implemented multidimensional data-analysis techniques and applied a "scene working set" approach, as well as static and dynamic video traffic models. We show that the behavior of a video sequence can be predicted with high accuracy by applying the *scene working set* technique. Applications of this technique for predicting bandwidth demands and allocating buffers in an ATM switch are also described in this study.**

***Index Terms*— Burstiness, communication systems performance, delay-sensitive traffic, multimedia communication, networks.**

I. Introduction

THE WORKLOAD of computer networks, ranging from the Internet to local intranets, has been increasing exponentially. Networks are used to transport various types of load which have widely varying resource demands, *quality of service* (QoS) requirements, and traffic patterns. The load generated by new types of communications services related to multimedia and video transmission is fast becoming one of the major sources of traffic in B-ISDN/ATM networks. The need for modeling this type of traffic is essential for addressing issues such as admission control and bandwidth allocation, and is also indispensable for network management (e.g., performance prediction, capacity planning, and optimization).

One of the problems that arises in modeling video-coded streams is that the bandwidth required is highly variable, both over time and in absolute value. Although using compression techniques does indeed effectively save bandwidth, it introduces extra nondeterministic factors in traffic behavior due to the variable amount of image redundancy that can be exploited. In fact, this type of transmission is usually referred to as *variable bit rate* (VBR) data stream transfer.

Manuscript received August 12, 1997; revised December 23, 1998; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor S. Tripathi. This work was supported in part by the PQE2000 project and in part by CESTIA CNR Research Center, Milan, Italy.

P. Manzoni is with the Universidad Politécnica de Valencia (DISCA), 46071-Valencia, Spain.

P. Cremonesi and G. Serazzi are with the Politecnico di Milano, 20133-Milan, Italy, and the Universitá della Svizzera Italiana, Lugano, Switzerland.

Publisher Item Identifier S 1063-6692(99)05392-3.

Several papers on modeling VBR video streams have been presented [1]–[10]. Most of them focused primarily on the identification of the statistical process(es) that best fit the empirical data of one [5], [6], [8], [10] or more [1], [3], [4], [7], [9] video sequences. In [8], Lazar *et al.* modeled the autocorrelation functions of the VBR bit streams with generalized stochastic transform-expand-sample (TES) methodology. Krunz and Tripathi [1] captured bit-rate variations at multiple time scales through a modulated process consisting of a second-order autoregressive process that varied around values derived by scene analysis. Lam *et al.* ([6], [9]) were the first to design algorithms for lossless smoothing.

The general applicability of the results obtained from analyzing a video stream is usually limited, since the best-fit parameters derived for a specific stream do not necessarily apply to other streams. In this paper, rather than investigating the adequacy of some stochastic processes to characterize a given pattern, we approach the traffic-load characterization problem from a global point of view, taking into account a population of 20 video streams. The suitability of a workload model depends upon the type of performance-evaluation study.

The load characterization may be approached at different levels by applying several techniques. We analyzed the statistical characteristics of a set of streams at frame level by considering each stream as an unordered set of frames (i.e., no temporal relationships between frames). The models obtained with this approach are also referred to as *static traffic models*. Frames are described by their resource consumption. By applying multidimensional data-analysis techniques, namely factor analysis and k-means clustering, we identified a small set of parameters describing the videos. We grouped the videos into three clusters based on this set. In the modeling studies in which only static characteristics are required, a video population can be represented by elements having cluster centroid parameters. The results obtained, i.e., the clusters (groups), can be useful in all situations where a video classification is required. Consider, for example, the design of a video-server. In this case, it is necessary to know how to organize the internal storage of the server and also to forecast which type of load possible clients will require from the server. This can be predicted if we know which cluster a video belongs to. Cluster analysis also provides us with the variables (i.e., the minimum and the maximum) that will be used by our algorithm—detailed in Section V-A—for forecasting the size of the next frame.

Analysis at the video-stream level showed that a simple characterization based on the subject of each video yielded

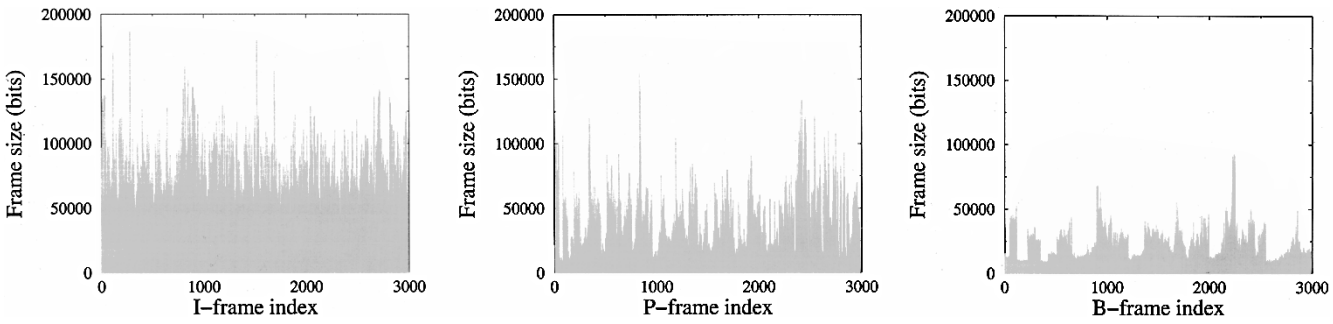


Fig. 1. Portions of *I*-, *P*-, and *B*-frame size sequences of the *race* video.

a classification that was similar to the one obtained when the resource consumption parameters were taken into account.

Knowledge of load fluctuations is required in performance studies in which the dynamic characteristics of the resource demands must be taken into account. The resulting models are also referred to as *dynamic traffic models*.

We investigated the dynamic characteristics of the videos using a simple technique that is applicable to all videos. First, we identified scenes. Then, we used a *scene working set* model to characterize the scene reference function of each video. The scene working set kept track of the scene sizes referenced during the last time interval of a given size. A scene was referenced when a frame with a similar bandwidth requirement was received. The idea behind the scene working set is directly related to the concept of scene: a scene referenced in the recent past has a high probability of being referenced in the immediate future. The scene references have a degree of *locality*.

In this paper, we show that the behavior of the experimental scene sequences can be predicted with a high level of accuracy by using a three-element last recently used (LRU) working set stack. Due to the intrinsically low complexity of our method, the prediction of the dynamic fluctuations of bandwidth demands—based on the working set model—can be performed on-line during the transmission of each video stream. Thus, no *a priori* scene behavior characterization is required. Furthermore, the classification of video streams, based on the number of scenes identified with the working set technique, matches the classification obtained with the resource oriented approach. As an application of the proposed characterization technique, we have applied the working set model to explore the performance of a buffer scheduler in an ATM switch.

This paper is organized as follows. In Section II, we analyze the experimental streams used. In Section III, we introduce the static characterization of VBR streams using factor analysis and a clustering algorithm at the frame level. In Section IV, we describe the characterization of the dynamic aspects of streams when applying the working set model at the scene level. Section V presents the results obtained using the working set model in the prediction of the bandwidth demands at the frame level and in the allocation of buffers of an ATM switch. Section VI concludes the paper.

II. Experimental Data

Let us initially subdivide the 20 traces used for our experiments into four groups, or categories, according to the subject of the video. The four categories, i.e., *movies, sports, news/talk shows*, and *various* were as follows.

- **Movies:** "James Bond: Goldfinger" (*bond*[1]), "Jurassic Park" (*dino*), "The Silence of the Lambs" (*lambs*), "Star Wars" (*star*), "Terminator II" (*term*), three episodes from the series "Mr. Bean" (*bean*), two episodes from "The Simpsons" (*simp*), an episode from "Asterix" (*aster*) and a 1994 movie preview (*movie*);
- **Sports:** ATP 1994 tennis final (*atp*), Hockenheim 1994 F 1 GP (*race*), 1995 superbowl final (*sbowl*), two 1994 soccer world cup matches (*soc1* and *soc2*);
- **News/Talk Show:** two German talk shows (*talk1* and *talk2*) and two German video conferences (*news1* and *news2*);
- **Various:** two MTV video clips (*mtv1* and *mtv2*).

The videos were compressed using an MPEG-1 compliant encoder. The quantization values were: $I = 10$, $P = 14$, and $B = 18$ using the pattern *IBBPBBPBBPBB*, which gives a group of picture (GOP) size of 12. Each MPEG video stream consisted of 40 000 video frames, which at 25 frames/s represented about 30 min of real-time full-motion video. Twenty frame-size sequences (FSS's) were derived [11] from the MPEG-1 compressed video streams, taking into consideration the size of each frame in bits.[2] The work described in this paper is based on analysis performed on the FSS's. Fig. 1 shows portions of the *I*-, *P*-, and *B*-frame size sequences of the *race* video.

While several parameters may change between the different MPEG encoders (e.g., the GOP pattern, the quantization level, and the image dimension), the meaning and the importance of *I* frames is common to the various standards. Indeed, there is a direct correspondence between *I*-frame size and the actual image size, while *B* and *P* frames are related only to the motion characteristics. As a result, *I* frames are usually the largest in size. For example, the average *I*-frame size for all 20 FSS's

[1] The names in italics are the short names used to indicate the corresponding video sequence.

[2] Thanks to Oliver Rose of the University of Würzburg, who made these traces. [Online]. Available FTP://ftp-info3.informatik.uni-wuerzburg.de/pub/MPEG/

TABLE I
DESCRIPTIVE STATISTICS OF THE I-FRAME SIZE FOR THE 20 VIDEO STREAMS CONSIDERED

category	*FSS*	*range*	*min*	*max*	*mean*	*stder*	*stdev*	*skew*	*kurt*
Movies	*bond*	222256	22336	244592	83297	451	26072	.55	.97
	dino	101976	17656	119632	55076	201	11639	.60	1.74
	lambs	119104	15120	134224	38023	221	12813	1.69	5.39
	star	111088	13728	124816	44012	244	14138	1.38	5.03
	term	63096	16464	79560	37387	144	8341	.57	.81
	movie	139104	14008	153112	57658	366	21138	.56	.84
	bean	135904	14272	150176	75161	336	19458	.91	1.05
	simp	133192	15304	148496	74045	333	19268	.13	.39
	aster	131800	15576	147376	71116	332	19180	.01	.17
Sports	*atp*	175768	15088	190856	75698	377	21801	1.13	4.25
	race	149256	36792	186048	79241	360	20826	1.06	1.44
	sbowl	126240	14600	140840	67952	330	19105	.43	.35
	soc1	164464	22712	187176	79142	437	25243	1.01	.67
	soc2	168032	22264	190296	70917	436	25228	1.27	1.84
News/ Talk Shows	*talk1*	70576	36192	106768	64734	176	10181	.42	.20
	talk2	97256	35496	132752	73773	225	13026	.78	.71
	news1	162640	31776	194416	85419	499	25604	1.03	1.35
	news2	176056	13832	189888	70616	363	20992	.53	1.57
Various	*mtv1*	179168	18520	197688	69861	418	24173	.89	1.17
	mtv2	234920	16488	251408	61396	439	25377	2.43	10.24

considered was 65742 bits, while the average sizes of *P* and *B* frames were 24588 and 10851 bits, respectively.

Thus, in problems dealing with resource reservation/allocation policies, the decisions based on *I*-frame size values are also valid for *P* and *B* frames. Furthermore, the *I*-frame sequence captures the dynamic characteristics of video traffic at a high level of granularity, since all the modifications of the data of an *I* frame produced by the motion are incorporated in the next *I* frame. The above considerations motivated the decision to concentrate our VBR traffic-modeling effort on the *I*-frame sequence only. The descriptive statistics for the *I*-frame sequence in each of the 20 FSS's considered are shown in Table I. To have an overall behavior description of the *I*-frame values, these include the range, minimum, and maximum. We then evaluated the first four moments of the distribution generated by the *I*-frame values. We obtained a central value (the *mean*), the dispersion around this value (the *standard deviation*), and two values evaluating the shape: the *skewness*, to determine the symmetry of the distribution, and the *kurtosis*, to determine the degree of peakedness [12].

III. STATIC CHARACTERIZATION OF VBR STREAMS

The modeling approach followed in this section considers each FSS as an unordered set of elements, i.e., *I*-frame sizes. Since *I* frames are dealt with as if no temporal relationships existed among them, we will refer to the models obtained with this approach as *static traffic models*.

In this framework, an FSS is characterized by a tuple of eight variables (see Table I) and is considered a point in the eight-dimensional space $\mathbb{R}^8$. The statistical properties of this data set are analyzed through multidimensional data-analysis techniques, namely factor analysis and clustering. Multidimensional (or multivariate) analysis is an extension of the classical mono-variate statistical analysis and is required since we want to take into consideration various variables at the same time to understand the relationship between them [13].

TABLE II
CORRELATION MATRIX $\boldsymbol{R}$ OF THE EIGHT STATISTICAL VARIABLES OF TABLE I

$\mathbf{R}$	range	min	max	mean	stder	stdev	skew	kurt
range	1.000	-.153	.983	.516	.878	.902	.415	.409
min	-.153	1.000	.028	.418	-.002	-.045	-.024	-.271
max	.983	.028	1.000	.599	.888	.904	.416	.363
mean	.516	.418	.599	1.000	.702	.691	-.227	-.358
stder	.878	-.002	.888	.702	1.000	.992	.229	.102
stdev	.902	-.045	.904	.691	.992	1.000	.229	.115
skew	.415	-.024	.416	-.227	.229	.229	1.000	.883
kurt	.409	-.271	.363	-.358	.102	.115	.883	1.000

TABLE III
MATRIX OF ESTIMATED FACTOR LOADINGS AND OF VARIMAX ROTATED FACTOR LOADING

	Estimated			Rotated		
variable	F_1	F_2	F_3	F_1^*	F_2^*	F_3^*
range	.967	-.127	.118	.924	-.309	.124
min	.002	.464	-.878	-.018	.680	-.991
max	.978	-.043	-.041	.931	-.300	-.056
mean	.648	.692	-.139	.756	.395	-.437
stder	.953	.179	.104	.975	-.040	-.014
stdev	.961	.159	.142	.983	-.044	.027
skew	.404	-.790	-.401	.168	-.958	-.057
kurt	.317	-.906	-.175	.089	-.951	.201

With factor analysis, we obtained the smallest number m of the considered eight variables that can be used to characterize each FSS. We then applied a clustering technique to identify the clusters of components (i.e., FSS's) having homogeneous characteristics. Each cluster can be represented in the traffic model by some of its members selected according to a suitable criterion. Factor analysis describes the covariance relationships among many variables in terms of a few underlying quantities called factors [13]. This technique consists of finding the eigenvalues in order of decreasing magnitude and the corresponding eigenvectors of the correlation matrix for the given set of variables. The relative size of each eigenvalue gives the variance of the corresponding factor, i.e., the eigenvector. Table II shows the correlation matrix $\boldsymbol{R}$ of the eight statistical variables considered in Section II.

There were three eigenvalues greater than one, respectively: $\lambda_1 = 4.41$, $\lambda_2 = 2.22$, and $\lambda_3 = 1.03$. Therefore, we considered three common factors, which accounted for a cumulative proportion of 0.957 $[(\lambda_1+\lambda_2+\lambda_3)/8]$ of the total standardized variance. The derived factor loading matrix is shown in Table III. Also shown in Table III are the factor values obtained with the varimax rotation criterion, a correction method that allows us to "spread out" the squares of the loading on each factor as much as possible.

From Table III, it can be seen that F_1 defines a group of the variables *range, max, stder*, and *stdev*, and F_2 defines a group of the variables *min* and *mean*. Since factor analysis considers negative values and values close to zero as irrelevant, factor F_3 will be excluded from the following.

Fig. 2 shows a scatter plot of F_1 and F_2 in the estimated and rotated cases. Both plots illustrate the presence of these two groups. We take *max* as a representative of F_1, since it has the highest load. We chose *min* as a representative of F_2 instead of *mean*, because it has the lowest correlation with *max* (0.028). We discarded *kurtosis* and *skewness*, since their values for factor F_1 were definitely lower than the ones of the four variables considered (*range, max, stder, stdev*), and they

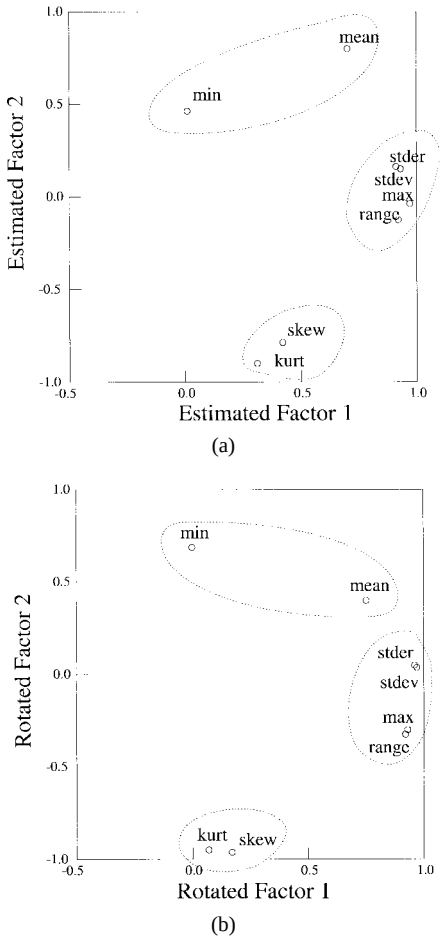


Fig. 2. Scatter plot of factors F_1 and F_2 in the (a) estimated (b) and rotated case.

assume negative values for factor F_2. Thus, two variables are sufficient to characterize the given data set, since they describe a large fraction of the total variance. We then proceeded with the grouping procedure, using the k-means clustering algorithm. The k-means clustering algorithm [14] subdivides a given data set into a specified number k of clusters in the space $\mathbb{R}^m$, with m being the number of variables that describe each element. Based on the factor analysis results, we have $m = 2$.

To estimate the optimal number k of clusters in which the given data set can be subdivided, we used two indicators of how well a partition works. For each variable, we used the overall mean-square ratio, i.e., a measure of the reduction of within-cluster variance between partition in k and $k+1$ clusters, and the ratio of the variance among the clusters and the within-cluster variance. Large values of the overall mean square ratio justify increasing the number of clusters from k to $k+1$. An optimal partition should also be characterized by values greater than one of the ratios of the variances among the cluster and within the cluster for each variable. These conditions are satisfied when we partition the given data set of FSS's into three clusters. Table IV shows the values of the variables *min* and *max*, corresponding to the centroids of the three clusters and the identifiers of the elements that belong to each cluster.

TABLE IV
CLUSTER CENTROIDS AND MEMBERS CORRESPONDING TO A PARTITION OF THE FSS INTO THREE GROUPS

variables	centroid of cluster 1	centroid of cluster 2	centroid of cluster 3
min	18946	19412	22997
max	130704	190909	248000
elements id	*aster, dino, lambs, movie, bean, sbowl, simp, star, talk1, talk2, term*	*atp, mtv1, news1, news2, race, soc1, soc2*	*bond, mtv2*
% of elements	55	35	10

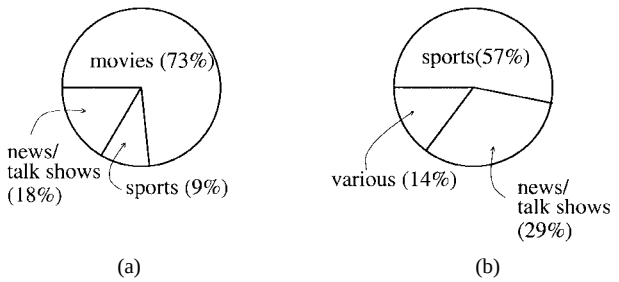


Fig. 3. Subject-oriented characterization of (a) cluster 1 and (b) cluster 2, obtained with a resource-oriented characterization (see Table IV).

A resource-oriented characterization of the three clusters is intuitive. The bandwidth requirement increases moving from clusters 1–3. While clusters 1 and 2, which represent 90% of the sample (see Table IV), contain the FSS that can be considered as "normal," the ones in cluster 3 are highly demanding FSS's having very large frame sizes. A subject-oriented characterization of clusters 1 and 2 is shown in Fig. 3. Cluster 3 is not shown, since its significance is low (having only two FSS belonging to *movie* and to *various*, respectively). As can be seen in Fig. 3, cluster 1 can be considered as being representative of *movie* streams and cluster 2 as being representative of *sports* streams. Indeed, 88.8% of the *movies* are in cluster 1, representing 72.72% of its elements, and 80% of the *sports* are in cluster 2, representing 57.14% of its elements. The four streams of *news/talk shows* are equally distributed between clusters 1 and 2, and represent a minor percentage of the elements of these two clusters, 18.18% and 28.57%, respectively. When new streams are considered, an FSS is assigned to a cluster if the values of its variables are within the ranges of the cluster's variables. When an element may be assigned to more than one cluster, it will be assigned to the cluster whose centroid is closest. The results obtained, i.e., the clusters (groups), can be useful in all situations where a classification of video is required. Consider, for example, the design of a video server. In this case, is necessary to know how to organize the internal storage of the server machine and also to forecast which type of load possible clients will require from the server. We can make these predictions if we know which cluster a video belongs to. Cluster analysis also provides us with the variables (i.e., the minimum and the maximum) that will be used by our algorithm—detailed in Section V-A—for forecasting the size of the next frame.

IV. DYNAMIC CHARACTERIZATION OF VBR STREAMS

In the approach described in Section III, the dynamic characteristics of a traffic stream were not taken into account and the temporal relationships of the *I* frames were ignored.

```
j := 1;
ref_f := f_I[1];
while(#t > 0) // while trace t has elements
    // looking for the 5 sub-traces s_k in t
    for k := 1 to 5 → eval (u_k, s_k, v_k | s_k subof t);
    // checking the additional condition for sub-trace s_5f
    if (abs(f_I[j+#u_i+#s_k −1] − ref_f) < T) → #u_5 := ∞;
    // selecting the sub-trace closer to the left part of t
    // that is, the sub-trace with component u_i with smaller cardinality
    eval (u_i, s_i, v_i | #u_i = min(#u_1, #u_2, #u_3, #u_4, #u_5));
    // evaluating scene-label's values for the (#u_i+#s_i − 2) I-frames in the window
    for k := j to (j+#u_i+#s_i − 2) → l_I[k] := (ref_f div T)*T;
    // reassigning value to ref_f
    ref_f := f_I[j+#u_i+#s_i −1];
    // evaluating scene-label's values for the I-frame at position (#u_i+#s_i − 1)
    l_I[j+#u_i+#s_i −1] := (ref_f div T)*T;
    j := j+#u_i+#s_i;
    // reassigning trace t
    t := v_i;
```

Fig. 4. The algorithm used for the generation of the sequence of SLS's.

However, when it is necessary to reproduce the same traffic-load behavior that has been found in a network during a given period of time, a suitable modeling technique must be considered. Models able to reproduce the dynamic characteristics of streams are referred to as *dynamic traffic models*. In this section, we describe a technique that allows the construction of dynamic models.

A. Modeling Video Behavior at Scene Level

The sequence of *I*-frame sizes (e.g., Fig. 7) can be segmented in *scenes*, such that the size of the frames of a segment are close in value. When we talk about scenes, we are not talking about what the typical spectator usually thinks of. Instead, a "scene" is considered as a block of frames that has certain size properties in common. Several algorithms have been applied to identify scenes (see e.g., [8], [5], [1]). We based our algorithm on these previous works, introducing slight modifications to reduce as much as possible the number of scene labels. Our algorithm is based on a set of five rules which apply a difference operator to the size of consecutive frames. Each scene is labeled by quantizing the size of its first frame at the granularity of a threshold value. We will see that, in the video streams considered, this technique identifies a limited number of scenes, while still preserving the dynamic characteristics of the original sequence.

The algorithm is based on a scaling function ($\mathcal{V}(i)$) defined as follows:

$$\mathcal{V}(i) = \begin{cases} 1, & \text{if } (f_I(i) - f_I(i-1)) \geq T \\ 2, & \text{if } -T < (f_I(i) - f_I(i-1)) < T \\ 3, & \text{if } (f_I(i) - f_I(i-1)) \leq -T \end{cases}$$

where $\{f_I(i)\colon i = 1, 2, \cdots\}$ is the *I*-frame size sequence and T is a threshold value. A trace t of values $\mathcal{V}(\cdot)$ is obtained from a sequence of *I*-frame sizes by applying the following algorithm:

$$\begin{aligned} &t := \langle\,\rangle; i := 0; \\ &\texttt{while } (\exists f_I(i)) \\ &\quad t := t\hat{}\langle \mathcal{V}(i)\rangle \\ &\quad i := i + 1; \\ &\texttt{wend;} \end{aligned}$$

An example of trace t is: ⟨1, 2, 2, 2, 2, 2, 1, 1, 1, 1, 3, 3⟩, which can be rewritten in a more compact form as $t = \langle 1\rangle\hat{}\langle 2\rangle^5\hat{}\langle 1\rangle^4\hat{}\langle 3\rangle^2$. The trace t is searched for five special subtraces. We define the function *subtrace of* (**subof**) as follows:

$$s \textbf{ subof } t \triangleq (\exists u, v.\, t = u\hat{}s\hat{}v) \wedge (\nexists p, q.\, u = p\hat{}s\hat{}q)$$

which means that if a subtrace exists, it is the one closest to the left margin of t. The five subtraces are identified using a sliding window of length $L_{\min} + 1$. $L_{\min}$ is a parameter representing the minimum length of a scene expressed in number of *I* frames.

Each one of these subtraces determines a scene change and activates the calculation of a scene label. The *I* frames that belong to the current window are replaced by the scene label. The final sequence of scene labels is called *scene-label sequence* (SLS).

Fig. 4 presents a code-based description of the algorithm used to scan trace t and generate the sequence of scene labels (l_I). Expression $\#a$ in the code refers to the length of trace a. Variable ref_f is the value of the last frame preceding a scene change.

The five subtraces used to determine a scene change are the following:

$$s_1 = \langle 1\rangle\hat{}\langle 2\rangle^{L_{\min}}; \quad s_2 = \langle 3\rangle\hat{}\langle 2\rangle^{L_{\min}}; \quad s_3 = \langle 1\rangle^{L_{\min}};$$
$$s_4 = \langle 3\rangle^{L_{\min}}; \quad s_5 = \langle 2\rangle^{5*L_{\min}}.$$

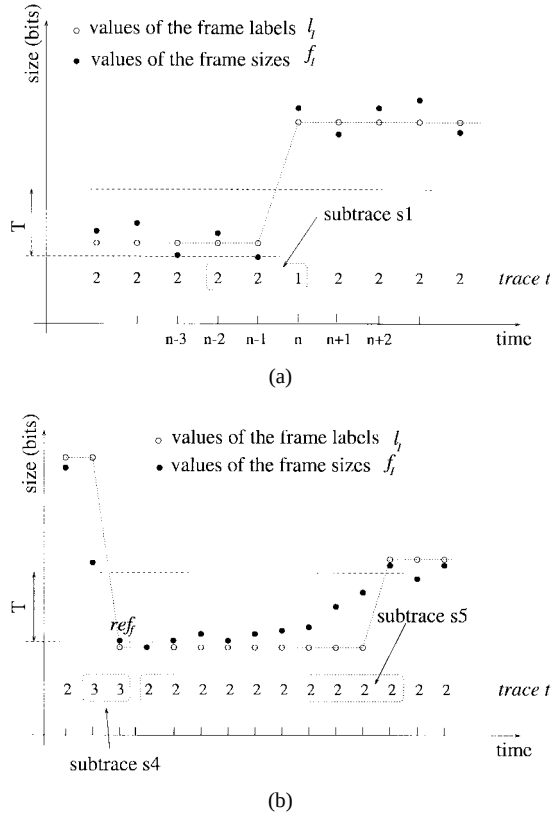


Fig. 5. Graphic representation of three different types of scene changes (described by subtraces 1, 4, and 5, respectively.)

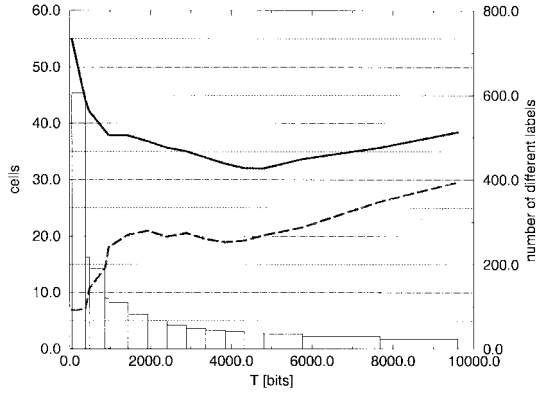


Fig. 6. Estimation of parameter T.

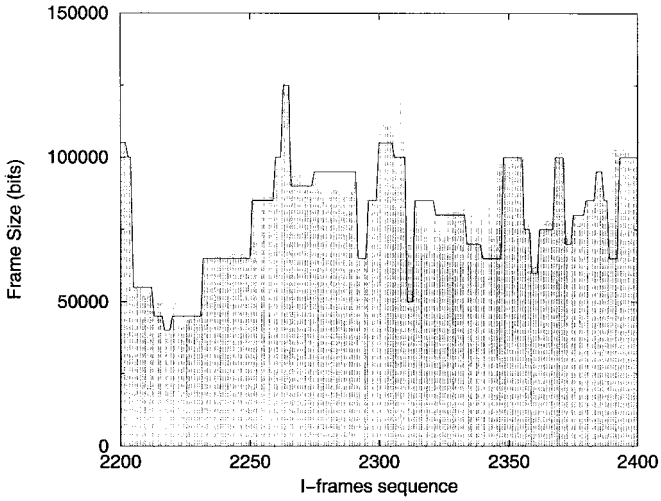


Fig. 7. I-frames plot (grey bars) and scene label sequence (solid line) of a segment of *aster*.

As shown in Fig. 5, subtrace s_5 requires that we check whether the difference of the value of the I frames that correspond to the first and last label is less than threshold T. Fig. 5 illustrates the identification of subtraces 1, 4, and 5, respectively. Subtraces s_1 and s_2 describe one abrupt scene change (positive and negative, respectively), while subtraces s_3 and s_4 represent a sequence of abrupt scene changes (positive and negative, respectively). Subtrace s_5 describes a possible scene change caused by a sequence of small differences between the sizes of successive frames. When the cumulative difference of frame size in the considered sequence is greater than threshold T, a scene change takes place.

When a new scene is identified, its frames are labeled with the size of the first frame at the granularity of the threshold value T. Several values of threshold T were investigated. Clearly, a tradeoff exists between the values of T and the number of scenes identified, and thus the accuracy of the SLS varies. We evaluated a range of possible values for T between 1–25 ATM cell payloads (an ATM cell payload corresponds to 48 bytes). Possible values for T were between 384 and 9600 bits. These values correspond to a typical ATM switch queue length. To evaluate T, we consider three factors: the number of different scene labels generated, the mean, and the standard deviation of the derived SLS with the correspondent FSS. Fig. 6 presents the result of this analysis. The dashed line represents the behavior of the mean, while the straight line shows the behavior of the standard deviation. The bars represent the amount of different scene labels generated. Position $T = 4890$ is where the standard deviation is minimum and the average is still an acceptable value. With such a value, the instabilities due to small fluctuations of frame size were avoided and the number of scenes identified was limited. At $T = 4890$, the number of different scene labels is smaller than 50, which is an adequate value to produce efficient implementation, as will be shown in Section V. To ease the following analysis, we rounded the value of T to 5000 bits. Since not all the frame-size jumps corresponded to scene changes, the minimum scene length was set to one second, i.e., $L_{\min} = 2$. Again, the value for $L_{\min}$ was chosen according to the definition of "scene" used in this work. A value of 1 s is a compromise which limits the overhead introduced at the switch (as will be detailed in Section V-B), while maintaining a good characterization of the FSS, as shown in Fig. 6.

Fig. 7 shows the actual values of *I*-frame sizes (grey bars) and the corresponding sequence of scene labels (solid line) of a fragment of *aster*. When our method was applied to the set of 20 FSS's with a threshold $T = 5000$ bits—which is only 7.5% of the mean frame size (65742 bits)—the average number of scenes identified was 24. The number of scenes of the 20 FSS's, grouped according to the three clusters obtained in the previous section, are shown in Fig. 8(a). It is interesting

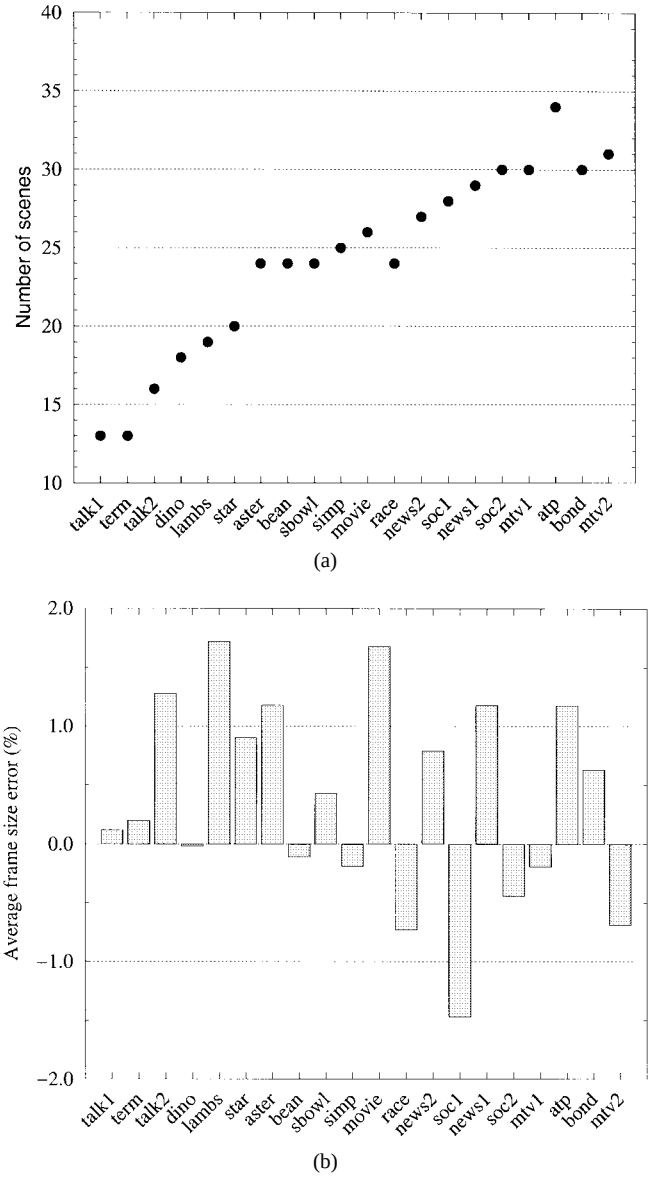


Fig. 8. (a) Number of scene labels for each stream. (b) Average error of frame size due to the use of scene label approximation.

to point out that when moving from cluster 1 to cluster 3, a trend in the number of scenes is evident (see Table V). This provides another way of characterizing the clusters with respect to the number of scenes, which is also related to their dynamism.

To evaluate the accuracy of the approximation of the SLS obtained using the algorithm in Fig. 5, we computed the average difference for each FSS between the actual frame size and the value of the corresponding scene label. Fig. 8(b) shows these differences (as percentages) for each FSS. As can be seen, these values, which can be considered as the errors introduced by the approximation of the SLS's, are all less than 1.8%, a very small value. The absolute error value averaged over all the 20 FSS's was 190 bits, while the absolute difference in the standard deviation was 656. In the following, we use the SLS's instead of the FSS's, since they are much easier to deal with and the percentage of errors they produce is negligible.

TABLE V
PARAMETERS USED BY FORECASTING ALGORITHMS

cluster	video	labels	p_1	p_2	p_3	p_r	Max label	Min label
1	*talk1*	13	0.9346	0.0113	0.0101	0.0440	100000	40000
	term	13	0.8872	0.0206	0.0230	0.0692	80000	15000
	talk2	16	0.9529	0.0113	0.0056	0.0302	125000	35000
	dino	18	0.9142	0.0137	0.0080	0.0641	105000	20000
	lambs	19	0.9382	0.0155	0.0065	0.0398	130000	15000
	star	20	0.9037	0.0215	0.0158	0.0590	125000	15000
	aster	24	0.8635	0.0107	0.0122	0.1136	135000	15000
	bean	24	0.9316	0.0083	0.0077	0.0524	145000	30000
	sbowl	24	0.8626	0.0155	0.0131	0.1088	140000	15000
	simp	25	0.8785	0.0158	0.0110	0.0947	140000	15000
	movie	26	0.8596	0.0128	0.0143	0.1133	155000	15000
	average	20.1	0.9024	0.0142	0.0181	0.0653	125000	21000
2	*race*	24	0.8650	0.0095	0.0128	0.1127	185000	35000
	news2	27	0.9181	0.0083	0.0074	0.0662	190000	15000
	soc1	28	0.8458	0.0086	0.0104	0.1352	180000	45000
	news1	29	0.9101	0.0064	0.0034	0.0801	175000	30000
	soc2	30	0.8572	0.0104	0.0080	0.1244	190000	20000
	mtv1	30	0.8596	0.0113	0.0092	0.1199	200000	20000
	atp	34	0.8800	0.0116	0.0110	0.0974	190000	15000
	average	28.8	0.8765	0.0094	0.0088	0.1053	187000	25000
3	*bond*	30	0.8641	0.0161	0.0116	0.1082	195000	25000
	mtv2	31	0.8815	0.0128	0.0098	0.0959	245000	15000
	average	30.5	0.8728	0.0144	0.0107	0.1020	220000	20000

B. The "Scene Working Set" Model

In this section, the problem of modeling and forecasting the dynamic behavior of scenes when applying the technique of the working set to the SLS's is studied. The working set technique has been used to model the behavior of a program in virtual memory systems. It characterizes the memory page references through the notion of *locality*. The original working set model has been reformulated by replacing the page numbers with the scene labels and identifying the *scene working set*. The scene working set keeps track of the scene labels referenced during the last interval of time. A scene label is referenced when a frame with the corresponding size is received. The idea behind the concept of locality is that a scene label that has been referenced in the recent past has a high probability of being referenced in the immediate future. This is directly related to the concept of scene. Very often, a scene changes into a new one with a similar bandwidth requirement; that is, references to a scene label tend to remain close in time.

A typical implementation of the working set model is the last recently used (LRU) stack model (SLRUM) [15], [16]. In our context, the LRU stack is a list of scene labels which are ordered according to the time of their most recent reference. The SLRUM is a generative model defined by a stationary stack distance probability vector

$$\underline{p} = (p_1, \cdots, p_k, \cdots, p_n)$$

where $p_i \geq 0$ and $\Sigma_{i=1}^{n}\ p_i = 1$. Each element p_i represents the probability that the next scene label will have the same value as the ith element of the stack.

To identify the scene working set, we used two vectors, r and l, with a number of elements equal to the number of different labels in each SLS. When a label was referenced, we looked for its position i in the vector l and incremented the ith element of r. To represent the stack structure, the vector l was reordered at each reference, so that the value of the last referenced label appeared in the first position. Table V shows the number of scene labels, the values of probabilities p_1, p_2, p_3, $p_r = 1 - \Sigma_{h=1}^{3}\ p_h$, and the maximum scene labels for

the 20 FSS's. Streams were grouped according to the clusters obtained in Section III and were ordered according to the number of scenes in each cluster. The average number of scene labels for clusters 1–3 was 20.1, 28.8, and 30.5, respectively. The *max* labels also increased from cluster 1 to cluster 3. Assuming the number of different scenes to be an index of the dynamism of a stream, we can say that streams of cluster 1 are less dynamic than streams of cluster 2, which in turn are less dynamic than those of cluster 3.

The analysis of the p_i values confirmed the presence of a working set with a limited size for all the FSS's, i.e., the scene label references were very local. For example, for the stream *talk2*, the probability that the next scene label is equal to the current one is $p_1 = 0.9529$, the probability that it is equal to one of the last referenced two different scene labels is $p_1 + p_2 = 0.9642$, and that it is equal to one of the last referenced three different scene labels is $p_1+p_2+p_3 = 0.9698$.

The average values of p_1 for clusters 1–3 were 0.9024, 0.8765, and 0.8728, respectively. These values agree with the characterization of the dynamism of the clusters we made before. The more dynamic a stream is, the less predictable the scene labels are. These results allow us to say that the static models obtained in Section III are also representative of the dynamic characteristics of the different streams. In the following section, we use the working set model to forecast the bandwidth requirements at the scene level.

V. Applications to Resource Reservation Problems

In this section, we want to validate our proposed methodology of workload characterization to real problems of resource management. We present two applications to the problems of bandwidth prediction of a video stream (in Section V-A) and to the evaluation of an ATM internal buffer utilization (in Section V-B).

A. Prediction of the Bandwidth Requirements

One of the most important problems to deal with when transmitting continuous media, like videos, is how much bandwidth we will require along the path [17]–[20]. This information can be used to evaluate aspects like: how many data-streams could share a certain path, which QoS can be provided, what the maximum bandwidth required to transmit a group of videos is, and so on. In this section, we describe a technique based on the application of the working set method to predict the bandwidth requirement at scene level of a video stream.

We implement a working set of size three with a stack of three elements. This stack is used to contain the values of the last three different scene labels. The next scene label was predicted by sampling the cumulative distribution of p_i with random numbers n_r, $0 \le n_r \le 1$. If $0 < n_r \le p_1$, then the next predicted scene label is equal to the one in the first stack position, i.e.,, to the last scene referenced. If $p_1 < n_r \le p_1+p_2$, then the next predicted scene label is equal to the one in the second stack position. If $p_1+p_2 < n_r \le p_1+p_2+p_3$, then the next predicted scene label is the one in the last stack position.

TABLE VI
Results of Four Bandwidth Forecasting Policies (at Scene Level) Based on a Working Set Model Applied on the 20 Video Streams

policy	cluster	predictions		
		correct %	overestimation %	underestimation %
conservative1	1	87.88	6.09	6.01
	2	86.05	7.05	6.89
	3	84.97	7.05	6.89
conservative2	1	82.05	9.10	8.85
	2	77.69	11.77	10.54
	3	76.54	11.61	11.85
pessimistic	1	81.47	12.95	5.58
	2	76.84	17.11	6.04
	3	76.56	16.90	6.54
optimistic	1	81.71	5.47	12.81
	2	77.03	6.34	16.63
	3	76.38	6.84	16.78

When $n_r > 1 - p_r$, we cannot predict any scene label by using the working set, i.e., a label fault has occurred. In this case, we propose four different prediction algorithms, namely: *conservative1*, *conservative2*, *pessimistic*, and *optimistic*. The basic criterion behind the two conservative approaches is to suppose that next frame will have the same size has the more recent ones. *Conservative1* algorithm makes its decision by taking into account the last label referenced, while the *conservative2* algorithm uses the average of the three labels in the working set. The *pessimistic* policy uses the maximum label value of the stream as the next label and finally the *optimistic* policy uses the minimum label value of the stream. The use of the maximum and the minimum of the scene label derives from the results of Section III.

The different forecasting policies correspond to different quality of services offered since the amount of bandwidth allocated decreases, moving from *pessimistic* to *conservative2*, *conservative1* and to *optimistic* policies.

The results of the application of the four policies to the 20 FSS's, grouped into the three clusters identified in Section III, are reported in Table VI. The number of times that the prediction was correct, as well as the number of times in which the scene label was over- or under-estimated are reported for each policy. As has previously been pointed out, the general trend of dynamism of the streams increases from cluster 1 to cluster 3. Indeed, in each policy, the number of times that a correct prediction is made decreases from cluster 1 to cluster 3; that is, the prediction is more difficult in more dynamic streams.

What is interesting to note is that a simple policy like *conservative1* allows for a bandwidth prediction which is correct 87.88% of the time for streams of cluster 1, the most consistent one. If the maximum QoS is required, we need to maximize the number of times in which enough bandwidth is allocated. The *pessimistic* policy satisfies this requirement 94.42% of the time, even if the bandwidth is overestimated 12.95% of the time. In spite of their simplicity, the worst result obtained from the four policies is a correct prediction 76.38% of the time.

B. Allocation of an ATM Switch Internal Buffer

We present in this section the results of simulation experiments which illustrate the impact of the proposed forecasting technique on the allocation of the internal buffer of an ATM

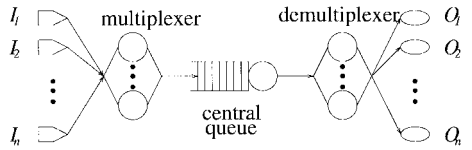


Fig. 9. Structure of the ATM switch model.

switch. Traffic control in ATM-based networks is the subject of an important research effort over the last few years. According to ITU-T Recommendation I.371 [21], the role of traffic control is to protect the network and the user in order to achieve predefined network performance objectives. Basically, traffic control refers to the set of actions taken by the network to avoid congestion. According to the I.371 standard, two functions are required for managing and controlling traffic in ATM networks: *Connection Admission Control* and *Usage Parameter Control*. Connection Admission Control refers to the actions taken by the network at call set-up phase in order to accept or reject an ATM connection. Usage Parameter and Network Parameter Control represent the set of actions taken by the network to monitor and control traffic on an ATM connection in terms of cell traffic volume and cell routing validity (this is usually called *police function*). There are two fundamentally opposite approaches for specifying traffic parameters: namely, a statistical approach and an operational (or algorithmic) approach. The latter is the one we are using in this paper. An operational approach defines the traffic parameters by means of a rule. The rule has been standardized in Recommendation I.371, and is called the generic cell rate algorithm (GCRA).

The technique we propose is meant to extend the virtual scheduling version of the GCRA. As will be shown later, our extension requires only a long-run average of an additional sum operation and a comparison for each incoming cell. To guarantee the required QoS, the crucial problem is to allocate the proper amount of network resources (i.e., in our case, buffer space). The most straightforward method would be to allocate a sufficient amount of buffer to satisfy the peak cell rate. This makes it possible to eliminate cell loss and delay variability. However, this approach wastes a high percentage of the internal buffer.

The queueing network model of an ATM switch with n I/O interfaces is shown in Fig. 9. We considered a central-queue approach based on [22], where the buffer is shared among all the input lines: each incoming cell is stored in the *central queue*. The internal buffer will be shared among two types of workload: video streams and other types of traffic (e.g., file transfers). While the former load usually has high QoS requirements, the latter can accept a "best effort" service. We apply the working set prediction technique (working set of size three) to determine the required amount of buffer space to be allocated for the video streams. The proposed allocation policy attempts to meet the QoS required from the video streams by minimizing the impact on the other types of traffic. In other words, we try to reserve enough buffer space to store video-stream cells with a proper QoS, limiting the waste of buffer space.

We model video streams using the ON–OFF model [23], [24]. In the ON–OFF model, the traffic load is composed of two distinct periods which alternate. During the ON period, cells arrive deterministically at intervals of duration τ. No cells arrive during the OFF period. In our case, the duration of the ON period (T_{ON}) depends directly on the size of the current *I* frame. The OFF period corresponds to the sum of the silent period between frames plus the duration periods of the *P* or *B* frames, up to the next *I* frame. The ON period length corresponding to the kth *I* frame can be calculated as follows:

$$T^i_{\mathrm{ON}}(k) = \frac{f_I(k)}{W_i} \tag{1}$$

where $f_I(k)$ is the size of the kth *I*-frame and W_i is the bandwidth of the ith switch input line. Since a switch is supposed to have n input lines, the worst-case situation is if we suppose that the n video streams reach the switch at the same time, which means that they are totally synchronized. In this case, the length of the period during which the largest buffer is required is

$$T_{\mathrm{ON}}(k) = \min(T^i_{\mathrm{ON}}(k)) \quad \forall i\colon 1 \le i \le n. \tag{2}$$

The required buffer space, in bits, is

$$q(k) = T_{\mathrm{ON}}(k) * \left(\sum_{i=1}^{n} W_i\right). \tag{3}$$

During its activity, our extended version of the GCRA algorithm, is required to calculate the actual size of the kth *I* frame for the n input lines. These values allow performance of the computation described up to this point.

The calculated value of the *I* frame allows us to forecast, using the algorithm described in Section V-A, the value of the next *I* frame (indicated as $f'_I(k+1)$). Therefore, by applying (1) using $f'_I(k+1)$ instead of $f_I(k)$ and then calculating (2), we can finally evaluate

$$q(k+1) = T_{\mathrm{ON}}(k+1) * \left(\sum_{i=1}^{n} W_i\right).$$

This value $(q(k+1))$ determines the maximum amount of buffer space required by n video streams. Fig. 10 shows the results of the application of this method on a switch with $n = 4$ I/O ports and the *conservative1* policy. Four sources were present, with each one transmitting the *aster* video stream in the switch. The average buffer utilization was $U = 30\%$. You can see how the reserved buffer size varies in time, trying to accommodate the incoming video traffic. The use of a peak-rate allocation policy would have required allocating an average value of 214 cells. With the *conservative1* policy, the average value of allocated cells was 168, saving about 27% of the buffer space. Since the proposed forecasting policy failed (in the sense that it underestimated the frame size) in about 6% of the cases, the evaluation of the correct buffer size also failed, giving a 1.36% rate of cell loss.

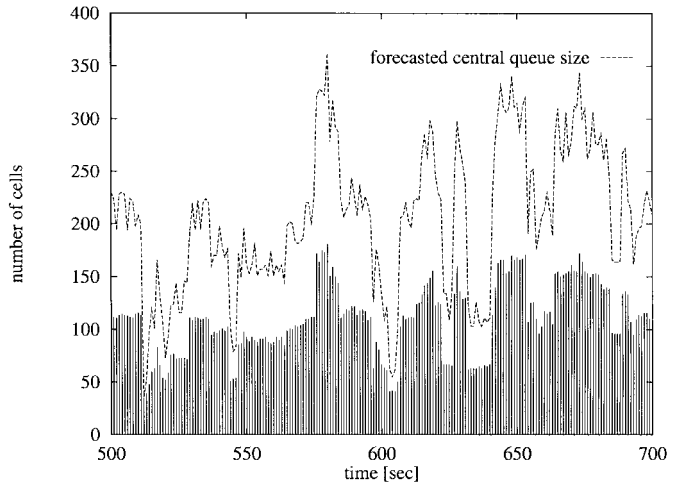


Fig. 10. Central buffer behavior with four synchronized inputs (*asterix* stream).

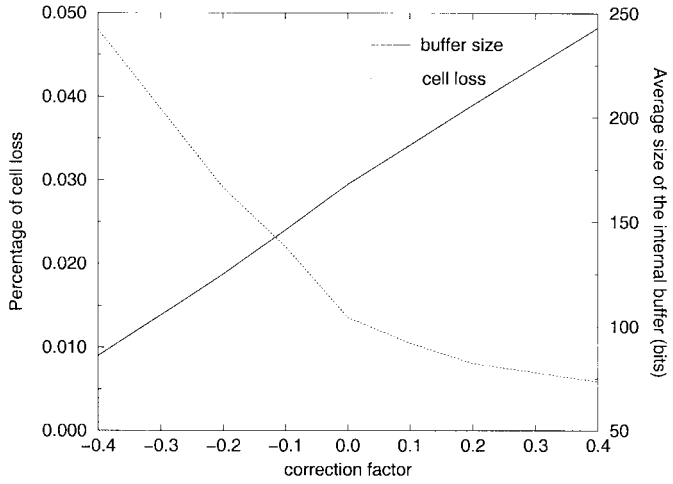


Fig. 11. Application of the correction factor α to $q(k+1)$.

If required by the QoS, we can reduce the cell loss by applying a correction factor α to $q(k+1)$, i.e., $q'(k+1) = (1+\alpha)*q(k+1)$. In this case, we obtain different combinations of cell loss and average buffer space, as shown in Fig. 11. By increasing the value of α, cell loss is reduced, but the average size of the internal buffer is increased. On the other hand, smaller values of α reduce the size of the buffer while increasing the percentage of cell loss.

VI. Conclusion

Design, optimization, and capacity-planning decisions all require predictions of network performance. To obtain accurate predictions, reliable models of both the workload and network are required. Indeed, workload characterization is a prerequisite to any performance study. In this paper, we have described several characterization techniques that can be applied to construct models of the load which is generated by coded video streams. Whereas conventional modeling studies were based on only one or very few empirical traces, our study takes 20 video traces into consideration. Resource-, subject-, and scene-oriented characterization approaches have also been used.

Multidimensional data-analysis techniques, factor analysis, clustering, and the scene working set technique are used to construct static and dynamic video traffic models. Applications of the proposed techniques for the prediction of the bandwidth demands of the 20 video streams, and for the allocation of buffers in an ATM switch, demonstrate the viability of the approach.

References

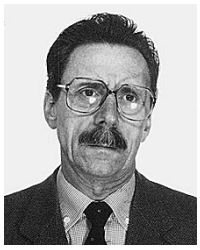
[1] M. Krunz and S. K. Tripathi, "Scene-based characterization of VBR MPEG-compressed video traffic," presented at Sigmetrics '97, Seattle, WA, June 1997.
[2] K. R. Krishnan and G. Meempat, "Long range dependence in VBR video streams and ATM traffic engineering," *Perform. Eval.*, vol. 30, July 1997.
[3] D. Heyman and T. V. Lakshman, "Source models for VBR broadcast-video traffic," *IEEE/ACM Trans. Networking*, vol. 4, pp. 37–46, Feb. 1996.
[4] D. E. Wrege and Jörg Liebeherr, "Video traffic characterization for multimedia networks with a deterministic service," presented at the IEEE INFOCOM '96, San Francisco, CA.
[5] M. Krunz and H. Hughes, "A traffic model for MPEG-coded VBR streams," presented at the Sigmetrics '95/Performance '95, Ottawa, Canada.
[6] S. S. Lam and G. G. Xie, "Burst scheduling: Architecture and algorithm for switching packet video," presented at INFOCOM'95, Boston, MA.
[7] E. W. Knightly and H. Zhang, "Traffic characterization and switch utilization using a deterministic bounding interval dependent traffic model," presented at the INFOCOM'95, Boston, MA.
[8] A. A. Lazar, G. Pacifici, and D. Pendarakis, "Modeling video sources for real-time scheduling," *Multimedia Systems*, vol. 2, no. 3, pp. 125–136, 1994.
[9] S. S. Lam, S. Chow, and D. K. Y. Yau, "An algorithm for lossless smoothing of MPEG video," presented at the ACM SIGCOMM'94, London, U.K.
[10] M. W. Garrett and W. Willinger, "Analysis, modeling and generation of self-similar VBR video traffic," in *Proc. ACM SIGCOMM'94*, London, U.K., pp. 269–280.
[11] O. Rose, "Statistical properties of MPEG, video traffic and their impact on traffic modeling in ATM systems," Inst. Comput. Sci., Univ. Würzburg, Tech. Rep. 101, Feb. 1995.
[12] M. G. Bulmer, *Principles of Statistics*. New York: Dover, 1979.
[13] R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1992.
[14] J. A. Hartigan, *Clustering Algorithms*. New York: Wiley, 1975.
[15] M. Kobayashi and M. H. MacDougall, "The stack growth function: Cache line reference models," *IEEE Trans. Computers*, vol. 38, pp. 798–805, June 1989.
[16] P. J. Denning and S. C. Schwartz, "Properties of the working-set model," *Commun. ACM*, vol. 15, no. 3, pp. 191–198, Mar. 1972.
[17] D. Anderson, R. Herrtwich, and C. Schaefer, "SRP: A resource reservation protocol for guaranteed-performance communication in the Internet," Int. Comput. Sci. Inst. (ICSI), Berkeley, CA, Tech. Rep. 90-006, Feb. 1990.
[18] A. Banerjea, D. Ferrari, B. Mah, M. Moran, D. Verma, and H. Zhang, "The Tenet real-time protocol suite: Design, implementation and experiences," Int. Comput. Sci. Inst. (ICSI), Berkeley, CA, Tech. Rep. 94-059, Nov. 1994.
[19] R. Braden and L. Zhang, *Resource reservation protocol (RSVP)—version 1 functional specification*, IETF Network Working Group, Nov. 1996.
[20] W. C. Feng, "Video-on-eemand services: Efficient transportation and decompression of variable bit rate video," Ph.D. dissertation, Univ. of Michigan, Ann Arbor, MI, Apr. 1996.
[21] *Traffic control and congestion control in B-ISDN*, International Telecommunication Union (ITU), Geneva, Switzerland, ITU-T Recommendation I.371, Aug. 1996.
[22] M. De Prycker, *Asynchronous Transfer Mode—Solutions for Broadband ISDN*, 2nd ed. New York: Ellis Horwood, 1993.
[23] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson, "Self-similarity through high-variability: Statistical analysis of Ethernet LAN traffic at the source level," *IEEE/ACM Trans. Networking*, vol. 5, pp. 71–86, Feb. 1997.
[24] V. Paxson and S. Floyd, "Wide-area traffic: The failure of Poisson modeling," *IEEE/ACM Trans. Networking*, vol. 3, pp. 601–615, June 1995.

Pietro Manzoni (M'97) received the M.Sc. degree in computer science from the Universite degli Studi, Milan, Italy, in 1989, and the Ph.D. degree in computer science from the Politecnico di Milano, Milan, Italy, in 1995.

He is currently an Assistant Professor of Computer Networks at the Universidad Politecnica de Valencia, Valencia, Spain. His research interests are in the theory, design, and implementation of communication protocols.

Paolo Cremonesi received the M.Sc. degree in aerospace engineering in 1992 and the Ph.D. degree in computer science in 1996, both from the Politecnico di Milano, Milan, Italy.

He is currently with the Politecnico di Milano, conducting research on performance evaluation of computer systems and networks, approximate methods for the solution of queueing network models, and high-performance computing.



Giuseppe Serazzi (M'95) received the Laurea degree in mathematics from the University of Pavia, Pavia, Italy.

He is a Professor of Computer Science at the Politecnico di Milano, Milan, Italy. His current research interests include workload characterization, modeling, and other topics related to the performance evaluation of computer systems and networks.