Multimedia documents Storage : An Evolutionary based Application

A. Vakali* **E. Terzi †** **L. Angelis*** **Mohand-Saïd Hacid ‡**

*Department of Informatics
Aristotle University
Thessaloniki, 54006 Greece

†Computer Sciences Dept.
Purdue University
W. Lafayette, 47906, IN- USA

‡Laboratoire d'Ingénierie des Systèmes d'Information
20, avenue Albert Einstein
69621 Villeurbanne - France

Abstract *Multimedia data storage is a critical issue in relation to the overall system's performance and functionality. This paper studies a multimedia document application which effectively guides data placement towards improving the quality of presentation of multimedia data. Several storage policies are proposed, towards better response and service times. Multimedia data dependencies, access frequencies and timing constraints are used to guide the storage policies under a certain representation model. The proposed placement policies are based on the simulated annealing algorithm and an extended improved version of the typical simulated annealing approach is presented. Experimentation results are presented and their impact on the total system's performance is commented and evaluated.*

Keywords: multimedia documents applications, multimedia data storage, evolutionary applications

1 Introduction

Multimedia data consist of various objects such as text, image, video and graphics (multimedia objects), which need to be synchronized and meet certain timing requirements. Physical storage of multimedia objects is a demanding and challenging problem due to the two principal constraints of multimedia data, namely size and timing. Representation models for multimedia data have been devised in order to represent the temporal relations among objects and specify the times at which discrete events occur. In [2, 3] a classification of the representation models, based on the notion of time is presented and the identified categories are: the *interval-based* and the *constraint-based* models. A number of different representation approaches have been also introduced in [2, 3] and they are classified into three other categories: *Graph Models*, *Petri-Net models* and *Object-Oriented Models*. Furthermore, another timeline model is proposed in [8] which captures user's interactivity on a set of multimedia documents.

Multimedia data are characterized by their timing constraints and their large storage requirements. Tertiary storage subsystems are quite appropriate to accommodate these data due to their large capacity and low cost. In [6] data placement policies on different technology tape libraries have been implemented. An optimal arrangement of cartridges and file-partitioning schemes are examined in [12] under a Carousel type mass storage system where organ-pipe arrangement is shown to be optimal. Furthermore, in [15] information placement schemes are considered based on three distinct models which correspond to three mid-range magnetic tape systems.

For the determination of the best storage policies we will use one of the simplest, yet powerful random search techniques for combinatorial optimization, namely the Simulated Annealing (SA) algorithm [10, 11]. In literature various versions of the SA algorithm have been presented for handling complicated optimization problems. Our proposed approach aims in the implementation of an improved version on the typical SA approach customized in our problem, the storage of multimedia data.

The structure of the remainder of the paper is organized as follows. In Section 2 the proposed representation model is introduced. The considered placement algorithms are presented in Section 3 where the typical simulated annealing as well as the improved version of the simulated annealing algorithm are described in detail. Section 4 includes the experimentation and the corresponding results. Finally, future work topics are discussed in Section 5.

2 Multimedia Data Representation

A multimedia application is based on the interaction and interconnection of multimedia objects. The user/client, in the considered multimedia applications, navigates (in an interactive way) through several multimedia objects which correspond to a set of Physical Objects. A set of various data "items" (called physical objects) are displayed at specific time intervals in a multimedia data stream. These data items are involved at specific time intervals in a multimedia data stream, they can be of different types and we store them in a storage medium as distinct physical entities (such as files or blocks). Thus,

Definition 1 : A *multimedia object* is defined as a set of tuples :

$$MultimediaObject = \{(mo_1, s_1, e_1), (mo_2, s_2, e_2), \ldots, (mo_n, s_n, e_n)\}$$

where mo_i represent the identity of the multimedia object i-th physical stored entity and s_i,e_i are the start and the ending times (respectively) of the i-th entity involved in a particular multimedia object.

Definition 2 : *Physical Object* is a specific data type entity which corresponds to a physical storage unit. A Physical Object may be involved in one or more multimedia objects and it is characterized by its start and end times specified with respect to the multimedia object they belong to.

The graph model is considered as quite appropriate to represent both multimedia data and the overall access pattern among different multimedia objects. A user "moves" from one object to another and this navigation can be captured by arcs in a directed graph the nodes of which correspond to distinct multimedia objects. The multimedia data representation model is based on the idea presented by the authors in [16, 17] and here the model considers a further representation analysis for each one of the nodes. The internal structure of each node is represented by a tree-like structure which is based on the concept of the timeline representation models. The two representation levels of our model are shown in Figure 1 and they are described below in further detail.

- **External level :** user's interaction can be represented by using the browsing graph as a "map" of places (multimedia objects) visited by the user. The *Browsing Graph* is a directed graph $G = (M, A)$ where $M = \{1, 2, \cdots, k\}$ is a set of k nodes corresponding to k multimedia objects and A is a set of directed edges connecting specific pairs of M. Each node in the browsing graph corresponds to a multimedia object itself, while the directed arcs represent the relationships among the various nodes (based on the idea of [4]). Any Browsing Graph can be uniquely defined by the so-called transition matrix which is a $(k \times k)$ matrix of access or transition probabilities , where by $P[i, j] = p_{ij}$, $(i, j \epsilon \{1, 2, \cdots, k\})$ we denote the probability of accessing node j from node i at a single step. It is obvious that the model we assume is a homogeneous Markov chain since the transition probabilities are time-independent. The notion of the "system" in the Markov chains terminology stands for the user's actions while the "states" of

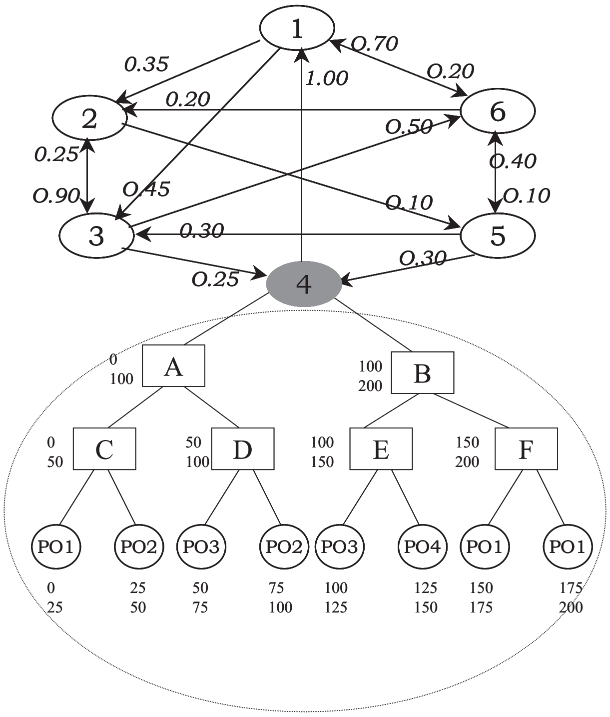

Figure 1: Graph-Tree Representation

the system are the multimedia objects or the nodes of the graph. It is also clear that the transition matrix P is a stochastic matrix, i.e. its elements are either zero or positive and its row sums are all ones, that is $\sum_{j=1}^{k} p_{ij} = 1$ for all i. Since P is the transition matrix of a homogeneous ergodic Markov chain, there is a unique vector $f = (f_1, \cdots, f_k)$, such that $\lim_{k\to\infty} P^k = f$ [7, 9].

Definition 3 : The row vector $f = (f_1, \cdots, f_k)$, where $\sum_{i=1}^{k} f_i = 1$ and

$$fP = f \quad or \quad f_j = \sum_{i=1}^{k} f_i P,$$

$$for\ j = 1, 2, \cdots, k$$

is called *vector of access frequencies* of the video objects $1, \cdots, k$ and its elements provide metrics to identify the popularity of each video object involved in the browsing graph.

- **Internal level :** At this level each node of the external level is further analyzed. A multimedia object can be considered as a collection of physical objects (i.e. storage units). Each physical object is considered as the *storage object* itself, and storage objects might have variable sizes and durations. Based on the idea presented in [8] and [13] a timeline object oriented approach is adopted in order to represent a single node of the external browsing graph. We introduce the *Time Segment Tree* model which allows us to emerge all the necessary temporal information needed to specify the synchronization constraints among the physical objects in a single multimedia object. The model is a timeline model because the starting and ending times of the physical objects displays are defined with respect to the absolute time of the specific multimedia object they belong to. The *Time Segment Tree* is a tree-like structure consisting of physical objects placed on the tree leaves, while their relations are indicated in internal nodes to specify the temporal relations between their child nodes.

It is obvious that a criterion should be adopted to "guide" the placement of the physical objects on the storage subsystem as follows :

Definition 4 : The *popularity of physical object x* in the Graph-Tree representation model is defined by

$$pop[x] = \sum_{i=1}^{k} f_i \times nr_{ix}$$

where f_i is the frequency of access (Definition 3) and nr_{ix} is the number of object x playouts in node i. It is obvious that the higher the frequency of access of a physical object the more popular this object is.

3 Multimedia Data Placement Algorithms

Tertiary storage systems (tape libraries in particular), are quite appropriate to accommodate voluminous multimedia data as they are characterized by high storage capacity. Recent technological advances have reduced the seek and service times of these devices and thus they can be seriously considered as an active part of the storage hierarchy (even in the case of multimedia data where certain timing constraints are imposed). Our data placement problem is to store N physical objects onto T tapes. We perform placement on tapes with Z zones. The considered data placement algorithms are of two main categories : *Constructive placement* and *Iterative improvement placement.*

3.1 Constructive Placement Algorithms

In the proposed storage policies the popularity of physical entities is the main criterion which guides the data placement. The main constructive placement algorithms are *organ-pipe* and *camel.* The organ-pipe and camel placement policies implemented within a tape consisting of Z zones are summarized as follows :
Organ-Pipe Placement

```
(A) Place the most popular object on the

    middle zone (Z/2) of the tape

(B) Allocate the next two popular objects

    on either side of the middle zone

(C) Go to (B) until all objects are placed.
```

Camel Placement

```
(A) Divide the tape into two consecutive

    tapes consisting of (Z/2) zones

(B) Implement Organ-Pipe placement on the

    two consecutive tapes (alternatively)

(C) Go to (B) until all objects are placed.
```

3.2 The typical Simulated Annealing (SA) Approach

Consider an arbitrary finite set Ω, called the solution space and an arbitrary function $C : \Omega \rightarrow R$, called the cost function $C(S) \geq 0$ for every $S \in \Omega$. Our aim is to find a global minimum of C in Ω. The SA algorithm generates a random sequence $S_n \in \Omega$ of feasible solutions which tends to converge as $n \rightarrow \infty$. The search begins from an arbitrarily chosen initial solution $S_{ini} \in \Omega$ and proceeds by generating and testing a sequence of solutions, each obtained as a random perturbation of the preceding one. The term "*perturbation*" means the move of the algorithm from any current solution S to a neighbor S_{try} through predefined legal operations. As search proceeds, any

```
T0 : Starting condition value;
R:Reduction Value for the condition  ( 0 <R < 1)
I : Number of perturbations (if no improvement of  Sbest occurs )

Sini ← initial Placement  ( randomly selected ) ;
T = T0 ;  S = Sini ;  Sbest = Sini ;
Repeat while STOP =  False ( Conditional loop )
        STOP = True ;   POINTER = 1
        Repeat while  POINTER ≤ I ( Perturbation loop )
                Stry  =  S
                C = C(S )
                Stry = perturb ( S )
                DC = C ( Stry ) - C ( S )
                if ( DC < 0 ) then
                        S = Stry  ( accept the improvement )
                        STOP =  False
                else
                        p = exp ( - DC/T )
                        u ← random number in U( 0,1 )
                        if ( u < p )  then
                                S =  Stry ( accept the wosrening )
                                STOP =  False
                        end if
                end if
                if ( C ( Stry ) < C ( Sbest ) ) then
                        POINTER = 1
                        Sbest = Stry
                else
                        POINTER ++
                end if
        end repeat
        if  ( STOP == False )
                T = T · R
        end if
end repeat
```

Figure 2: The general structure of the SA Algorithm

time a perturbation of the current solution S leads to an improvement of the cost function, (i.e. if $C(S_{try}) < C(S)$), it is accepted as the basis for the next perturbation.On the other hand, if a perturbation of the current solution leads to a worsening (i.e. if $C(S_{try}) > C(S)$), it is accepted with probability $p = \exp\left(-\frac{\Delta C}{T}\right)$ where , $\Delta C = C(S_{try}) - C(S)$ and T is an algorithm parameter called "*temperature*". The temperature is gradually reduced during the process, through a decreasing sequence $T_n$such that $T_n \to 0$ called the cooling schedule. The algorithm continuously keeps record of the best solution and the random walk passed from and this is denoted by S_{best} After a predefined number of perturbations in a standard temperature without improvement of S_{best}, the temperature is lowered according to a cooling schedule and the search continues. The most common cooling schedule, the so-called exponential, is defined by $T_{n+1} = T_n \cdot R$, where $0 < R < 1$ is a reduction factor, usually close to one. The algorithm needs also a stopping criterion, so we usually agree to terminate the search when after a predefined large number I of perturbations at a standard temperature no improvement of the current solution occurs. On termination of the process, the solution S_{best} is reported as optimal.

The solution space of our data placement problem is the set of all possible permutations of the N physical objects $\{PO_1, PO_2, ..., PO_n\}$. As cost of each permutation P we considered the expected service time (EST) defined by the formula:

$$EST(S) = \sum_{i=1}^{N}\sum_{j=1}^{N} pop[i]pop[j](s_j s_{rate} + t_j t_{rate})$$

where i,j refer to the current head location (i) towards the requested location (j). Notice that s_j and t_j are the number of bytes to search and transfer (respectively), while s_{rate} and t_{rate} are the search and transfer rates (respectively). Finally, N refers to the total number of storage entities. Therefore, the simulated annealing placement commences with an initial placement determined by a constructive placement procedure and is repeatedly modified in search for cost reduction. The general structure of the algorithm (as it is implemented for the specific data placement problem) are shown in Figure 2. The initial placement can follow either one of the constructive placement methods (camel, organ-pipe placement) , or a random policy. Another important issue is the *perturbation function* followed since this function will rearrange the physical objects within the tapes. Three perturbation functions have been implemented on our model :

1. *Circular Shift - CS* : The perturbation function moves PO_n to the position of the tape currently occupied by PO_1 while for every $i \neq n$ PO_i is moved to the position of PO_{i+1}.

2. *Single Random Selection - SRS* : a physical object is selected randomly and moved to a random location on the tape, the rest of the objects are moved and placed after the chosen object. If current placement is $(PO_1, \ldots, PO_{k-1}, PO_k, PO_{k+1}, \ldots, PO_n)$ and we select object PO_k to be moved to position 1, we get the permutation $(PO_k, PO_1 \ldots, PO_{k-1}, PO_{k+1}, \ldots, PO_n)$.

3. *CS + SRS*: a combination of Circular shift and Single Random Selection where at each iteration either Circular Shift or Random Selection is implemented with equal probability for each one.

3.3 An Improved Version of the Simulated Annealing

Here we proposed an improvement SA approach which is based on the idea of [1] such that the SA algorithm is repeatedly executed for a prefixed number of times N_{total}. The total number of trials is distributed to blocks of equal length N_{block}. When a block of trials is executed, the process checks which is the minimum service cost up to that point and calculates its relative frequency f_{best} in the specific block. Then, f_{best} is compared with a predefined (small) probability level $0 < f_{low} < 1$. If $f_{best} \leq f_{low}$, the values of the parameters I and R (presented in Figure 2) are increased and the search continues with the next block. If $f_{low} < f_{best}$, the search proceeds to the next block with the same parameters. Every B blocks, i.e. every B N_{block} repetitions of the SA algorithm, the process calculates the relative frequency f^*_{best} of the best design up to that point and makes a comparison with a (large) probability level $0 < f_{high} < 1$. In case $f^*_{best} > f_{high}$, the procedure stops and the best placement found is reported. Since the parameters I and R are continuously increased, the process becomes gradually slower. It is therefore necessary to fix reasonable initial values to the procedure's parameters to restrict the duration of the process.

4 Experimentation - Results

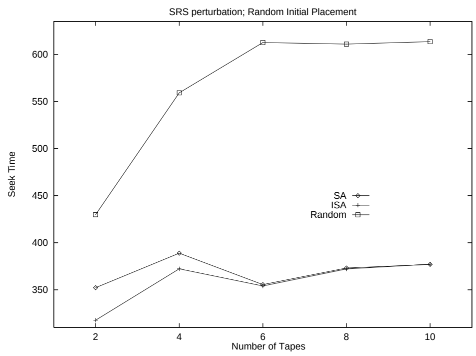

Figure 3: SRS+Random Initial Placement (Seek Time)

We have run various experimentations under workloads for zoned tapes since they are widely used and they tend to replace PBOT tapes. Numerous sets of requests were generated in order to evaluate the previously described placement schemes, based on the idea

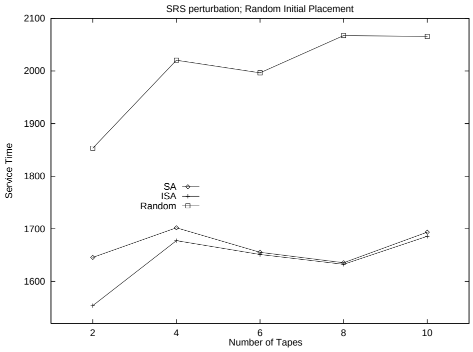

Figure 4: SRS+Random Initial Placement (Service Time)

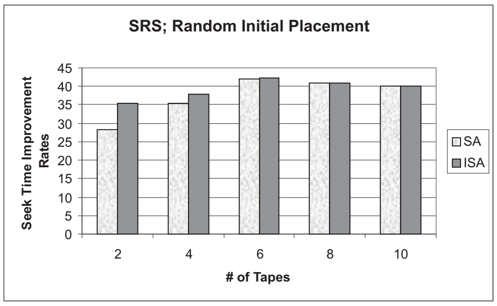

Figure 5: SRS+Random Initial Placement (Seek Time)

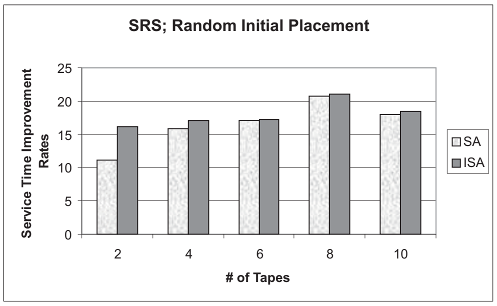

Figure 6: SRS+Random Initial Placement (Service Time)

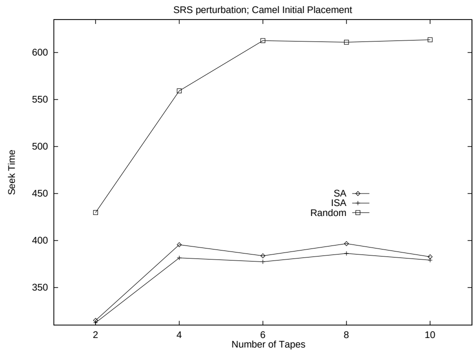

Figure 7: SRS+Camel Initial Placement (Seek Time)

of the simulated annealing algorithm and they are compared to random placement. The artificial workload of the video objects has been generated as follows : the total number of the physical objects of the pool increases with the number of nodes of the browsing graph; the number of physical objects each node contains is uniformly distributed between 1 and the total number of objects in the pool; each physical object's size varies from some hundreds of KB to hundreds of MB. It is obvious that the total size of multimedia objects is equivalent to the size of real multimedia data. Simulation results refer to both seek and service time and the workload was generated such that a large percentage of the total tape space is occupied. More specifically, for storage systems containing small number of tapes (4, 2 tapes) 75-90% of the total available storage capacity is occupied. This percentage inevitably decreases when the number of tapes increases, as the workload is constant. This approach allowed experimentation on the system's performance when the stored objects are either scattered among the available tapes or stored on a small number of them.

Figures 3, 4, 7 and 8 compare the service and seek time values resulted by the implementation of simple and improved simulated annealing as well as random placement, under specific rearrangement functions and initial placement algorithms and for a storage system with a varying number of tapes $(2 \cdots 10)$. Improved simulated annealing proves to be more beneficial than simple simulated annealing. Figures 5 and 6 show the improvement rates of the

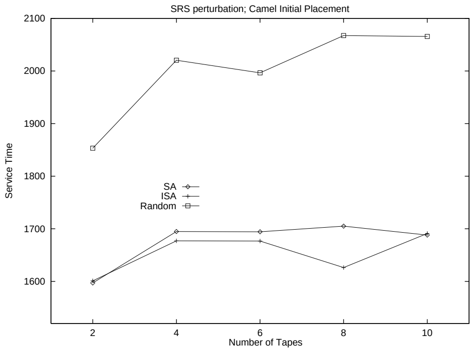

Figure 8: SRS+Camel Initial Placement (Service Time)

implementation of simulated annealing-based placements in comparison to random placement. These diagrams show that the improvement rates of service (seek) time of the simple SA algorithm reach 40% (20%) respectively, while the corresponding improvement rates for the ISA algorithm are 43% (23%).

5 Conclusions - Future Work

This paper considers a timeline model for multimedia objects representation towards effective data storage. The proposed model considers two representation levels such that the external is supported by a structure called the Browsing Graph and the internal level is based on a time-segment tree structure that represents each multimedia object separately. The physical object is a term used to guide an initial storage policy which is appropriately modified by the proposed SA and ISA algorithms. Experimentation results have indicated that both SA and ISA show significantly beneficial performance as compared to the random placement whereas most of the ISA experimentation cases prove to be more advantageous than the corresponding SA in terms of both seek and service times.

Further research could extend the proposed model to implement the SA algorithm under a different criteria for perturbations acceptances and/or under a different cost function $C()$ which as an example could be based on the specific workload. An extension of the algorithm to adapt another optimization idea based on all implementations of [1] is currently under development.

References

[1] L. Angelis, E. Bora-Senta and C. Moyssiadis : Optimal exact experimental designs with autocorrelated errors through a simulated annealing algorithm, *Journal of Computational Statistics and Data Analysis*, 2000, to appear.

[2] Elisa Bertino and Elena Ferrari : Temporal Synchronization Models for Multimedia Data, *IEEE Transactions on Knowledge and Data Engineering*, Vol.10, No.4, 1998.

[3] Yong-Moo Kwon,Elena Ferrari, Elisa Bertino : Modeling spatio-temporal constraints for multimedia objects, *Knowledge and Data Engineering*,(30), pp.217-238, 1999.

[4] Yie-Tarng Chen : Physical storage model for interactive multimedia information systems, PhD Thesis, Department of Electrical Engineering, Purdue University, 1993.

[5] A.L. Chervenak: Tertiary Storage-An Elevation of New Applications, PhD Dissertation, University of California at Berkeley, 1994.

[6] Stavros Christodoulakis, Peter Triantafillou and Fenia Zioga : Principles of Optimally Placing Data in Tertiary Storage Libraries, *23rd International Conference of Very Large Databases (VLDB)*, pp.236-245, Athens, Greece 1997.

[7] D.R. Cox and H.D. Miller : *The Theory of stochastic processes*, Chapman and Hall, 1977.

[8] N.Hirzalla, Ben Falchuk, and Ahmed Karmouch : A Temporal Model for Interactive Multimedia Scenarios. *IEEE Multimedia* 2(3): pp.24-31, Fall 1995.

[9] D.L. Isaacson and R.W. Madsen : *Markov chains theory and applications* John Wiley and Sons, 1976.

[10] Kirkpatrick, S., Gelatt, C. D., and Vechhi, M. P. : Optimization by simulated annealing, *Science Journal*, 220, (1983), 671-680.

[11] Metropolis, N.A, Rosenbluth, A., Rosenbluth, M., Teller, A. and eller E. : Equation of state calculations by fast computing machines, *Journal of Chem. Physics*, 21, (1953), pp.1087-1092.

[12] S.Sesardi, D. Rotem and A. Segev : Optimal Arrangements of Cartridges in Carousel Type Mass Storage Systems, *The Computer Journal*, Vol.37, No.10, pp.873-887, 1994.

[13] Cyrus Shahabi : Scheduling the Retrievals of Continuous Media Objects, *PhD Thesis*, Computer Science, University of Southern California, 1996.

[14] V.S. Subrahmanian : *Principles of Multimedia Database Systems*, Morgan Kaufmann Publishers Inc., 1997.

[15] A.Vakali and Y.Manolopoulos: Information Placement Policies n Tertiary Storage Systems, Storage Models for Multimedia Object, *Proceedings of the Hellenic Conference of New Information Technologies*, pp.205-214, Oct 1998.

[16] A.Vakali, E. Terzi and A. Elmagarmid : Representation models for Video Data Storage, *Journal of Applied System Studies, Special Issue on Distributed Multimedia Systems with Applications*, to appear 2001.

[17] A.Vakali and E. Terzi : Video Data Storage Policies : An Access Frequency Based Approach, *Computers and Electrical Engineering Journal*, to appear 2001.