PERGAMON

Computers and Electrical Engineering 28 (2002) 447–464

Computers and Electrical Engineering

www.elsevier.com/locate/compeleceng

Video data storage policies: an access frequency based approach

Athena Vakali *, Evimaria Terzi

Department of Informatics, Aristotle University, Box 114, 54006 Thessaloniki, Greece

Received 20 July 2000; accepted 25 September 2000

Abstract

Video applications are characterized by their increased requirements for huge storage spaces and timing synchronization. Video data storage is a critical issue due to the so-called I/O bottleneck problem in relation to the quality of service while accessing video applications. The main contribution of the paper is that it considers video data dependencies, access frequencies and timing constraints in order to introduce a video data representation model which guides the storage policies. Two video data representation levels are considered to capture the frequencies of accesses at external (video objects) and internal (video clips) levels. A simulation model has been developed in order to evaluate the placement strategies. Video data placement is performed on a tertiary storage subsystem by both constructive and iterative improvement policies. Iterative improvement placement has been proven to outperform the other video data placement approaches.

Keywords: Multimedia data storage; Video data processing; Video data representation; Video physical layout; Data placement algorithms

1. Introduction

Video data are characterized by timing relations and constraints imposed by users interactions. The main objective of video applications is to support a hiccup free display of video data. If only disk-based secondary storage systems are used to store and manage this huge amount of data, the system cost would be extensively high. A tape-based storage system seems to be a reasonable solution for lowering the cost of storage and management of continuous data. The most important design issues of a video storage server is to provide jitter-free video services as well as to

* Corresponding author. Tel.: +30-31-99-8415; fax: +30-31-99-6360.
E-mail address: avakali@csd.auth.gr (A. Vakali).

0045-7906/02/$ - see front matter
PII: S0045-7906(00)00068-9

promote utilization of the storage bandwidth to accommodate more users. In Ref. [3] the issues of placing continuous objects on a tape-based tertiary storage system and also the ensuing replacement and scheduling policies have been considered. A new performance model for a video storage server, which takes into account striping strategies, disk scheduling policies, data placement schemes, block sizes, buffer requirements and initial delay time simultaneously, is proposed in Ref. [22]. The configuration of a storage system that supports different video data types is studied in Ref. [14], whereas in Refs. [17,18] the configuration of a system to support sharing for continuous media type is studied. Placement techniques and scheduling algorithms that guarantee continuous display of objects within a heterogeneous disk storage system are introduced and evaluated in Ref. [24]. Tertiary storage devices and tape libraries in particular, have been pointed out in Refs. [5,6,15]. The physical structure of multimedia data storage subsystems is described in Ref. [10]. It is interesting to note that tertiary systems have different and diverse performance factors not applicable to all technologies as indicated in Ref. [15]. Data placement has been studied for tertiary storage subsystems, in Refs. [6,16,23]. Iterative improvement placement algorithms, and simulated annealing in particular, have also been implemented in Ref. [4]. A detailed description of the simulated annealing algorithm is given in Ref. [21] while its implementation on database systems has been discussed in Ref. [20].

Representation models for multimedia data have been introduced to represent the temporal relations among objects and specify the timing at which discrete events occur. In Refs. [1,2] a classification of the representation models, based on the notion of time is presented. The main classes been discriminated are the *timeline*, the *interval-based* and the *constraint-based* models. Also, a number of different representation approaches have been also introduced in Refs. [1,2,7] and classified into three main categories: *graph models*, *Petri-net models* and *object-oriented models*. In Refs. [9,11] video objects representation models are categorized into *stream-based models* and *structured models* with respect to their physical requirements and from the perspective of the database management system. In Ref. [19] a different multimedia data representation model is proposed under a considered database schema based on a hierarchical tree structure for video objects representation. Furthermore, a new timeline model is proposed in Ref. [12] which captures user's interactivity on a set of multimedia documents.

This paper presents a model for video data placement on a considered tertiary storage subsystems, under specific video data representation. The paper's main contribution is related to the following key issues:

- The *navigation path* among various video objects is considered for video data representation and their storage. The relationships among various video objects capture the user's access patterns on a considered multimedia storage server. These access patterns are considered to guide the storage policies and video data are no longer considered to be independently accessed.
- The *physical objects* corresponding to the video clips (included at a particular video data object) define the actual storage units involved in the data placement policies. Then, the frequency of access of a particular video clip specifies the popularity of the corresponding storage unit such that the data placement favors the most popular frequently accessed video clips.

The structure of the remainder of the paper is organized as follows. In Section 2 the proposed representation model is introduced. Section 3 provides a description of the data placement criteria

and both constructive and iterative improvement placement algorithms are further discussed. The main modules of the simulation and the storage system model are briefly described in Section 4. Experimentation and results are given in Section 5 with discussion and result comments. Finally, conclusions and future work topics are discussed in Section 6.

2. Video data representation model

2.1. Video data representation – preliminaries

A video application is based on the interaction and interconnection of video objects. For example, in a video movie application each user/client navigates (in an interactive way) through several video objects which correspond to a set of video clips. Video clips consist of video frames which are the last level of granularity in video structuring. Some definitions (in relation to the model's definition) follow:

Definition 1. Video clip is a sequence of video frames. Each video clip has its own size, duration, presentation rate and is stored as an entity in files or blocks of the storage system.

Definition 2. Video object is a set of n video clips characterized by temporal constraints. A video object is defined as a set of tuples:

$$\text{video object} = \{(\text{vc}_1, s_1, e_1), (\text{vc}_2, s_2, e_2), \ldots, (\text{vc}_n, s_n, e_n)\}$$

where vc_i represents the identity of the ith video clip belonging in the video object and s_i, e_i are its start and the ending times (within this particular video object).

Definition 3. The browsing graph is a directed graph $G = (N, A)$ where $N = \{1, 2, \ldots, k\}$ is a set of k nodes corresponding to k video objects and A is a set of directed edges connecting specific pairs of N. Additionally, every edge in A is weighted by an access or transition probability. Any browsing graph can be *uniquely* defined by the so-called transition matrix.

Definition 4. The transition matrix P associated with the graph G, is a $(k \times k)$ matrix of access or transition probabilities, where p_{ij} $(i, j \in \{1, 2, \ldots, k\})$ denotes the probability of accessing node j when currently being at node i at a single step.

Definition 5. A vector $f = (f_1, \ldots, f_k)$, where $\sum_{i=1}^{k} f_i = 1$ and $fP = f$ or $f_j = \sum_{i=1}^{k} f_i p_{ij}$ for $j = 1, 2, \ldots, k$, is called *vector of access* frequencies of the video objects $1, \ldots, k$ and its elements provide matrices to identify the popularity of each video object involved in the browsing graph.

Theorem 1. If P is the transition matrix of a homogeneous ergodic Markov chain, then there is a unique vector $f = (f_1, \ldots, f_k)$, such that

$$\lim_{k\to\infty} P^k = f$$

Proof. A thorough study and classification of finite Markov chains and the proof of this theorem is given in Refs. [8,13]. The theorem gives us a way of approximately evaluating the access frequencies of the nodes, by simply calculating powers of the transition matrix. Theorem 1 gives a way to evaluate the relative frequency of accessing (retrieving) nodes $1, \ldots, k$ respectively in a long run, based on the transition probabilities of the initial browsing graph. It is known that in the theory of stochastic processes the vector f is called the equilibrium or stationary distribution of the Markov chain since any element represents the limiting probability of accessing the node i after infinite number of steps (in the long run). □

2.2. The nested-graph representation model

In our case a graph model has been considered to represent video data as well as the overall access pattern among different video objects. The basic idea of the proposed model is depicted in Fig. 1. A user "moves" from one video object to another and this navigation can be represented as arcs in a directed graph. The nodes of the graph correspond to distinct video objects. For example, such a seven-node graph is depicted in Fig. 1 for a set of seven video objects. The model considers a further representation analysis for each of the nodes since each node consists of a number of video clips. The structure of the internal node is represented by a graph model as well.

The nodes of this second graph correspond to the video clips (participating in the video object) while the arcs represent user's navigation preferences within the video object. It is obvious that the assumed model 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 the system are the video 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

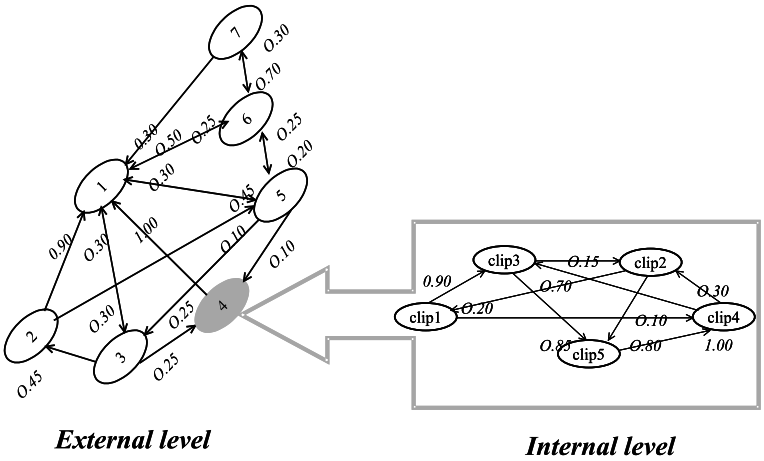

Fig. 1. The nested-graph representation model.

and its row sums are all ones, $\sum_{j=1}^{k} p_{ij} = 1$ for each i. Thus, two representation levels will be supported in our model:

- *External level:* User's interaction can be represented by using the browsing graph as a "map" of nodes (video objects) visited by the user. Each node in the browsing graph corresponds to a video object itself, while the directed arcs represent the relationships among the various nodes (based on the idea of Ref. [4]).
- *Internal level:* At this level each node of the global view structure is further described. Since timing relations and constraints arise among clips within a node, there is a need to focus on the node content with respect to clips relationships over time and users' access pattern. A video object (i.e. a movie) can be represented as a collection of video clips. Each video clip is considered to be the actual *storage object* and storage objects may have variable sizes and durations. Notice that a clip may be part of more than one video objects. These storage objects are members of continuous media data and should be retrieved and displayed at a prespecified continuous rate. Based on the idea of Ref. [17], a video object is represented by a graph model as well. This graph is called the *internal browsing graph*, is a directed weighted graph (Definition 3) and each node corresponds to a video clip itself, while the directed arcs represent the relationships among the various clips. Therefore, a vertex in the graph represents a link from one video clip to another.

The external level browsing graph principles, still hold for internal level browsing graph of a single video object. Therefore, the internal browsing graph of a node (video object) x is related to a transition matrix P_{int}, (as the transition matrix P, Definition 4), the elements of which correspond to the transition probabilities among the n clips of video object x.

Definition 6. The vector of access frequencies of video clips will have as elements $v_{x,j}$ which define the popularity of clip j within node x. This clip popularity vector has the same properties with the video objects access frequency vector f (Definition 5).

The above described (two level) model will be referred to as *nested-graph* representation model. Thus, the *nested-graph* representation model refers to a two-level representation model with browsing graphs used to represent video data relationships and constraints both in the external and the internal levels. A complete example of a nested-graph model is depicted in Fig. 1.

3. Video data placement algorithms

3.1. The placement criteria

Certain criteria are needed to guide the placement of video clips in order to propose effective video data physical layout. As mentioned in the previous section, the video objects include a number of video clips which specifies a "pool" of physical objects. The same video clip may be included in more than one video objects. However, only one copy of each video clip is kept in the storage system. Each video clip should be placed effectively and appropriately to guarantee high quality of service.

Definition 7. The popularity of a video clip j in the nested-graph representation model is defined by

$$\mathrm{pop}[j] = \sum_{i=1}^{k} f_i v_{ij}$$

where f_i is the frequency of access (Definition 5) of the video object corresponding to node i (estimated by the formula given in Theorem 1), and $v_{i,j}$ is the popularity of video clip j within node i (Definition 6). The popularity value of a video clip is higher when the clip is part of a "popular" node and when it should be played for several times within this node. Therefore, all objects included in popular nodes have a high popularity value.

The popularity of video clips within the nested-graph representation model is the basic criterion which will be used to guide the implementation of the data placement algorithms on a tertiary storage subsystem. Tertiary storage systems and tape libraries in particular, are quite appropriate to accommodate voluminous video data as they are characterized by high storage capacity. Furthermore, 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 video data where certain timing constraints are imposed. Our data placement problem is to store C video clips 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*.

3.2. Constructive placement

The *organ-pipe* and the *camel* placement algorithms are considered as indicative policies under the constructive placement approach. Fig. 2 presents these algorithms as applied in a tape library storage system with T tapes.

The organ-pipe and camel placement policies as implemented within a tape consisting of Z zones are summarized next:

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 alternatively on the two consecutive tapes
(C) Go to (B) until all objects are placed.

3.3. Iterative improvement

The iterative improvement placement, commences with an initial placement determined by a constructive placement procedure and is repeatedly modified in search for cost reduction. If a

```
pool[1..C]        // Pool of C video clips
pop[1..C]         // Popularity of the C video clips
notstored []      // array index to possible non-stored clips

VC[1..C] ← sorted pool [ ] array in decreasing pop[ ] values

i ← 1
j←1
while (there are still Unallocated Clips and free Tape Space )
{
   STORE i-th clip in (i mod T) -th tape at first available segment.

   if (the clip is stored)
      i ← i+1
   else if (space is not enough)
       STORE i-th clip on the next tape with adequate free space.
       if (the clip is stored)
              i ← i+1
       else  // there is no tape with enough space available
              notstored[j] ← i
              j ← j+1
              i ← i+1
}
```

Fig. 2. Constructive placement.

modification results in a reduction, it is accepted (otherwise it is rejected). This algorithm is based on simulated annealing algorithm which is a popular algorithm for combinatorial optimization used in our placement problem. A description of the algorithm is given in Fig. 3. The *initial placement* of the physical entities within the tapes can follow either one of the constructive placement methods described above, or a random policy.

The rearrangement *(perturbation)* of the clips within the tapes can be applied following two strategies.

1. Interchange: Select two clips of the current configuration randomly, and interchange their positions.
2. Rotation: Make a left (or right) circular shift of current configuration.

The expected service time is an indicative measure of the cost of the placement indicated by the end of each perturbation, and is evaluated by the following formula:

$$\text{expected service time} = \sum_{i=1}^{N}\sum_{j=1}^{N}\text{pop}[i]\text{pop}[j](s_j s_{\text{rate}} + t_j t_{\text{rate}})$$

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.

```
L0 : Starting condition value;
R:Reduction Value for the condition  ( 0 <R  < 1)
Npertub : Number of perturbations (if no improvement of  Sbest occurs )

Sini ← initial Placement  ( randomly selected ) ;
L = L0 ;  S = Sini ;  Sbest = Sini ;
Repeat while STOP =  False ( Conditional loop )
        STOP = True ;  POINTER = 1
        Repeat while  POINTER ≤ Npertub ( Perturbation loop )
            Stry  =  S
            cost = cost(S )
            Stry = perturb ( S )
            Dcost = cost ( Stry ) - cost ( S )
            if ( Dcost < 0 ) then
                S = Stry  ( accept the improvement )
                STOP =  False
            else
                p = exp ( - Dcost/L )
                u ← random number in U( 0,1 )
                if ( u < p )  then
                    S =  Stry  ( accept the wosrening )
                    STOP =  False
                end if
            end if
            if  ( cost ( Stry ) < cost ( Sbest ) ) then
                POINTER  =  1
                Sbest  =  Stry
            else
                POINTER ++
            end if
        end repeat
        if  ( STOP == False )
            L = L · R
        end if
end repeat
```

Fig. 3. Iterative improvement algorithm.

4. The simulation model

4.1. Simulator's modules

Our simulation model consists of the three main modules (Fig. 4):

- *The data representation module:* The browsing graph is constructed based on users access patterns. Representation is performed in both internal and external levels and the access frequen-

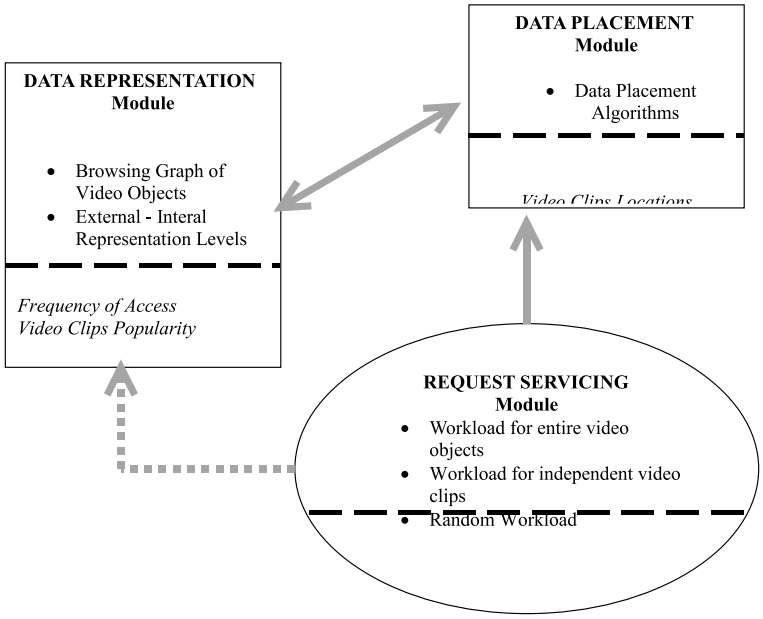

Fig. 4. The modules of the simulation model.

cies are evaluated by using the formula given in Theorem 1. The video clips' popularity estimation is based on Definition 7.

- *The data placement module:* One of the data placement algorithms (described in Section 4) is applied according to the selected tape topology and the location of each physical object is identified.
- *The request servicing module:* The request workload refers to specific nodes of the external browsing graph. When such a request arrives, the video clips that correspond to the video object been pointed by the user are retrieved from the tapes on which they are stored on. These clips are elevated on the cache memory such that the request could be serviced. Requests referring to a number of video clips that do not necessarily belong to the same video object as well as randomly created requests can also be cases for further experiments.

4.2. Simulator's tertiary storage model

The tertiary storage subsystem used for our simulation studies is a tertiary storage library. These libraries come in many different sizes and configurations and despite their significant differences, they all contain: *drives, robot arms* and *tapes or disks*. Our library is thought to have one robot arm, which is assumed to be capable of moving between any tape stored in the library. The robot arms are involved in:

- *Pick and place:* A pick operation refers to the robot picking up a tape from its storage space in the library and taking it into the drive.
- *Move:* A move operation refers to the robot moving between different locations on the shelf.

Although there is variation in the times required for these operations, they are generally modeled as constant times. This is acceptable because the variation is much smaller than the total time required to exchange tapes or disks on drives:

$$Robot\_arm\_service\_time = Pick\_time + Move\_time + Put\_time$$

The drives, which are also assumed to be identical, perform the following operations: seek, rewind, read, write, load and eject. The seek and rewind operations for tapes are modeled as constant startup times followed by a constant transfer rate. Therefore the tape access time is defined by the times involved in the servicing main actions:

$$Drive\_service\_time = Rewind\_T + Eject\_T + Load\_T + Seek\_T + Transfer\_T$$

Therefore the total access time for a tape operation which includes a tape switch operation is defined as follows:

$$Total\_service\_time = Robot\_arm\_service\_time + Drive\_service\_time$$

The tapes of the tape library are thought to be linear surfaces divided into a certain number of fixed-size *segments*, which are the smallest accessible parts of the tape. *Sections* consist of a number of consequent segments, while *tracks* consist of a number of consequent sections. The number of segments that will be allocated to a data object mainly depends on the object's and the segment's size. Thus the number of segments s reserved for a single stored clip is given by

$$s = \left\lceil \frac{\text{size of object}}{\text{size of segment}} \right\rceil$$

If the object's size is not exactly divided by the segment's size, then the last segment allocated to an object is not full, and therefore some space remains empty.

5. Experimentation – results

We have run the simulator under workloads for Zoned tapes as long as 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. Simulation results refer to both seek and service time. More specifically, random, organ pipe, camel and iterative improvement placement strategies have been implemented and the system's performance has been estimated for a system with a varying number of tapes $(2, \ldots, 10)$ and a constant number of nodes of the external browsing graph and vice versa (Figs. 5–8).

The artificial workload of the video objects been stored has been created as follows:

- the total number of clips of the pool increases with the number of nodes of the browsing graph;
- the number of clips each node contains is uniformly distributed between 1 and the total number of clips in the pool;
- each clip's size varies from some hundreds of KB to hundreds of MB.

It is obvious that the total size of video objects is equivalent to the size of real video data. Furthermore, the workload was created in a way that a large percentage of the total tape space to

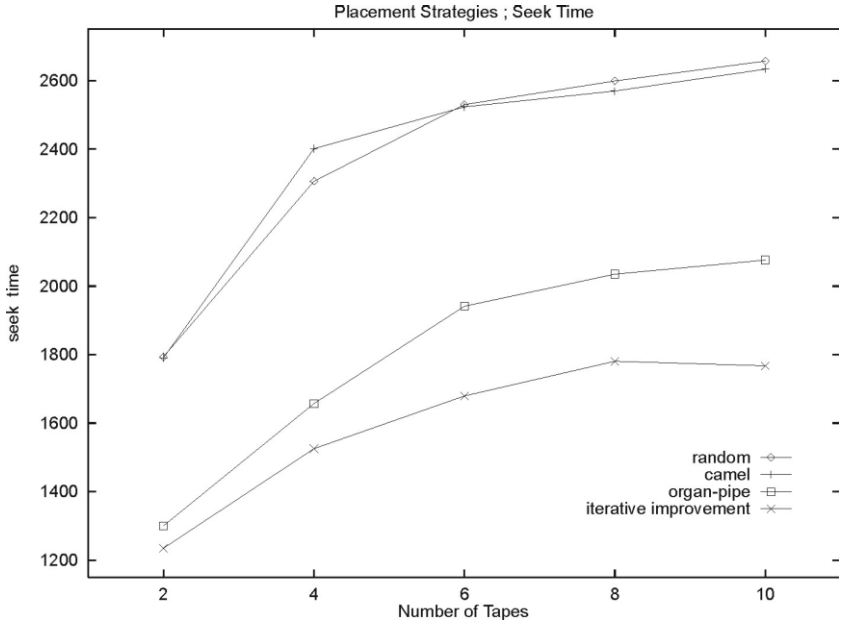

Fig. 5. Seek time versus number of tapes.

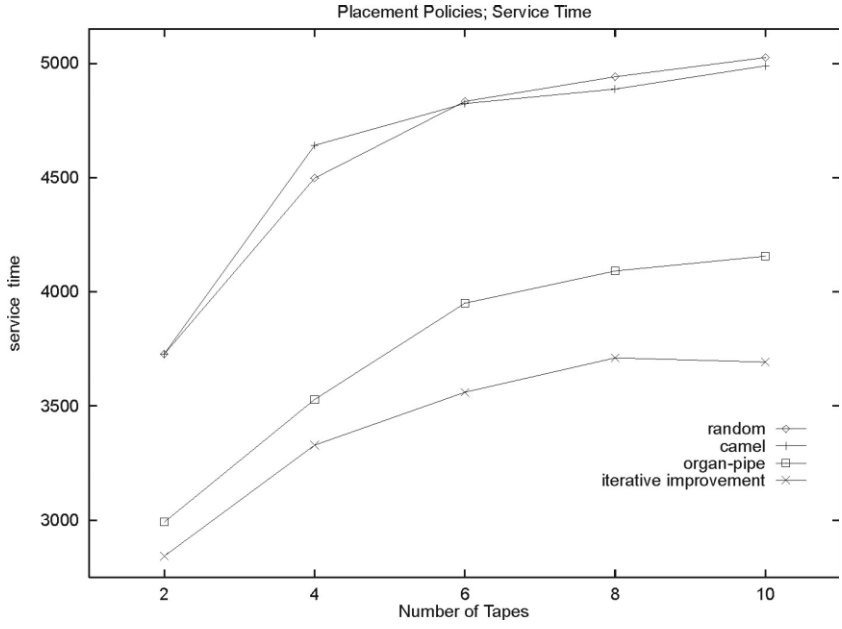

Fig. 6. Service time versus number of tapes.

be occupied. More specifically, for storage systems containing small number of tapes (2,4 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 us to

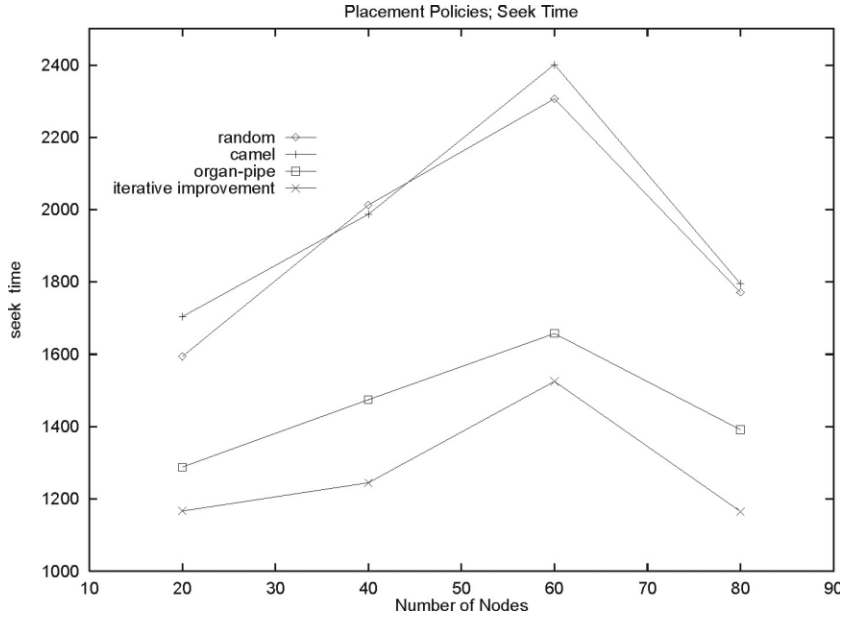

Fig. 7. Seek time versus number of nodes.

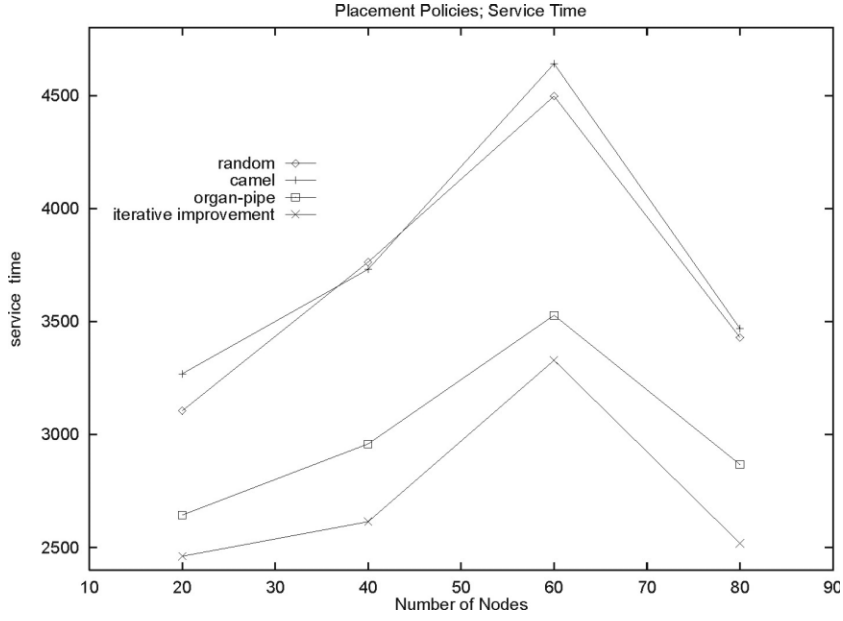

Fig. 8. Service time versus number of nodes.

experiment on the system's performance when the stored objects are either scattered among the available tapes or stored on a small number of them and therefore to lead up to conclusions relative to the impact of seek and tape exchange times to the total system's performance.

The results indicate that iterative improvement considerably improves system's performance. Both seek and service times increase with the number of tapes, which can be easily explained since the larger the number of tapes the more tape exchanges (and therefore higher delays). The number of nodes of the external browsing graph affects the system's performance, however its influence does not follow a specific pattern since the final values of the expected seek and service time depends of the internal structure of the video objects. We should notice that organ pipe placement scheme proves to be better than both camel and random placement. Camel placement has not been proved beneficial to the performance improvement since it has resulted in worse seek and service times than random placement.

Finally, Figs. 9–14 show the system's performance when organ-pipe, camel and simulated annealing placement policies are applied. These figures refer to the expected service time for systems with varying number of tapes $(2,\ldots,10)$ and multimedia applications with constant number of external graph's nodes and vice versa.

Figs. 5–8 are further discussed, since they are indicative for the placement policies relative performance. More specifically, Figs. 7 and 8 show that service (seek) time obtained by camel placement worsens system's performance by a rate reaching even 5% (7%). On the other hand, organ pipe placement significantly improves system's performance since the respective service times obtained are better than those obtained by random placement by a percentage ranging from 15% (for external graph consisting of 20 nodes) to 22% (for 60-node external graph). The respective improvement rates that are related to seek time vary from 19% (20-node external graph) to 28% (60-node external graph). Finally, the placement indicated by the implementation of the iterative improvement algorithm leads to further improvement of the system's performance since the service (seek) times obtained are better than those obtained by random placement at an

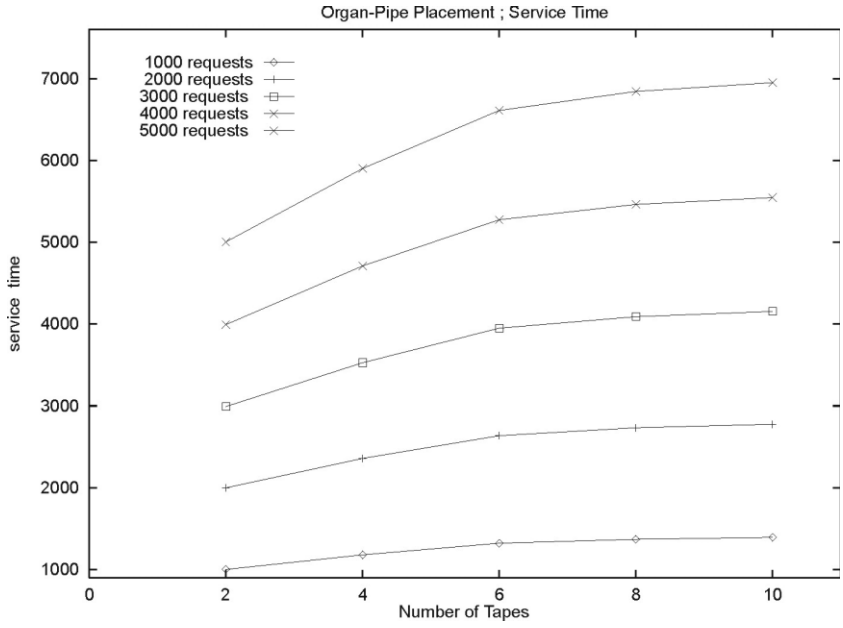

Fig. 9. Organ-pipe versus number of tapes.

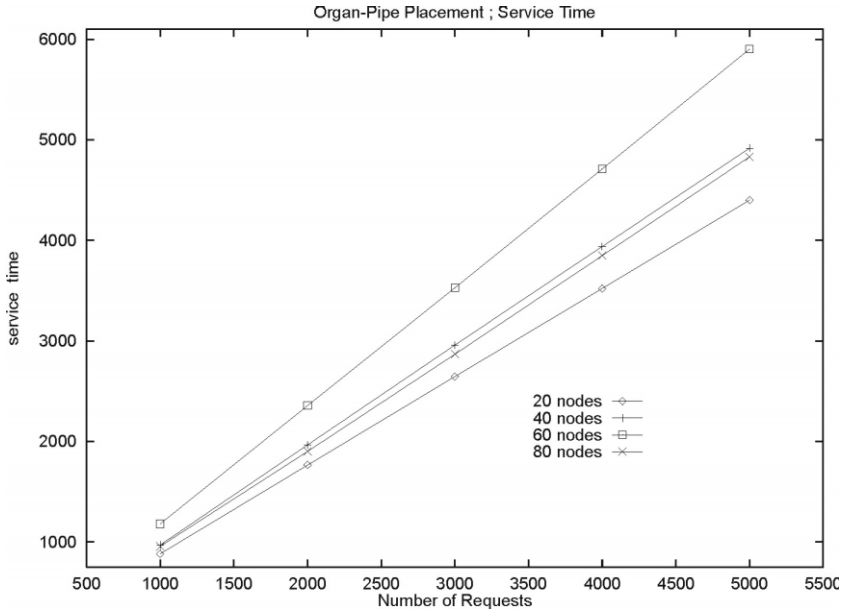

Fig. 10. Organ-pipe versus number of nodes.

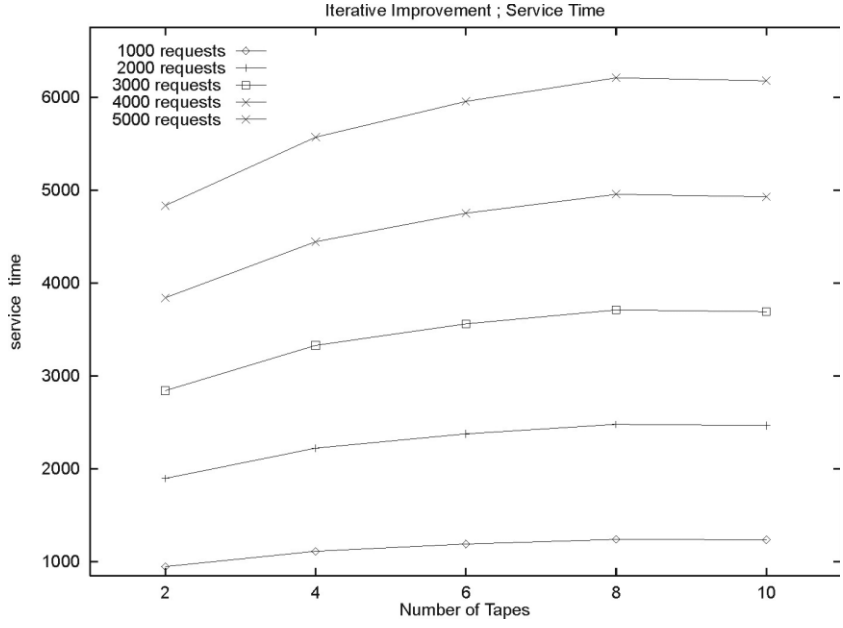

Fig. 11. Iterative-improvement versus number of tapes.

average rate of 26% (33%). Table 1 summarizes the resulted improvement percentage for service and seek times under the different storage polices, as compared to the random placement.

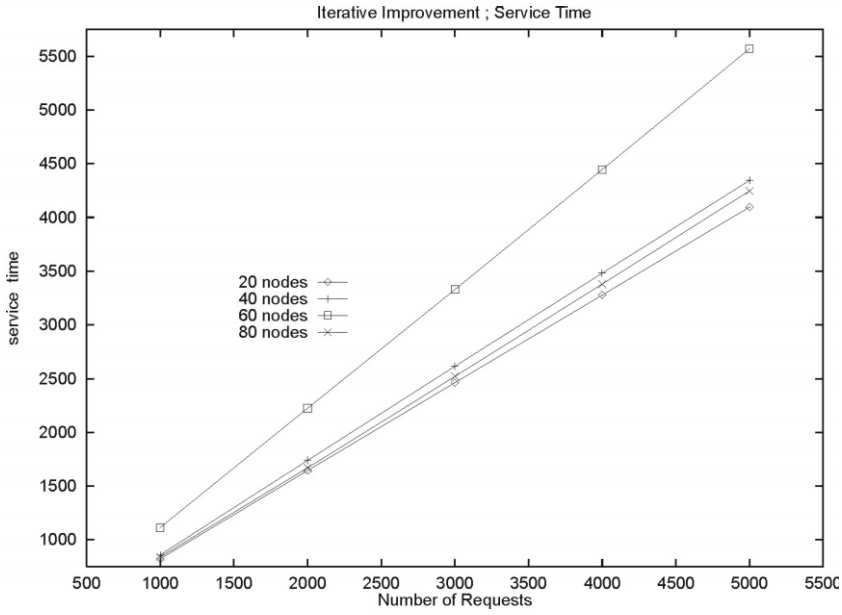

Fig. 12. Iterative improvement versus number of nodes.

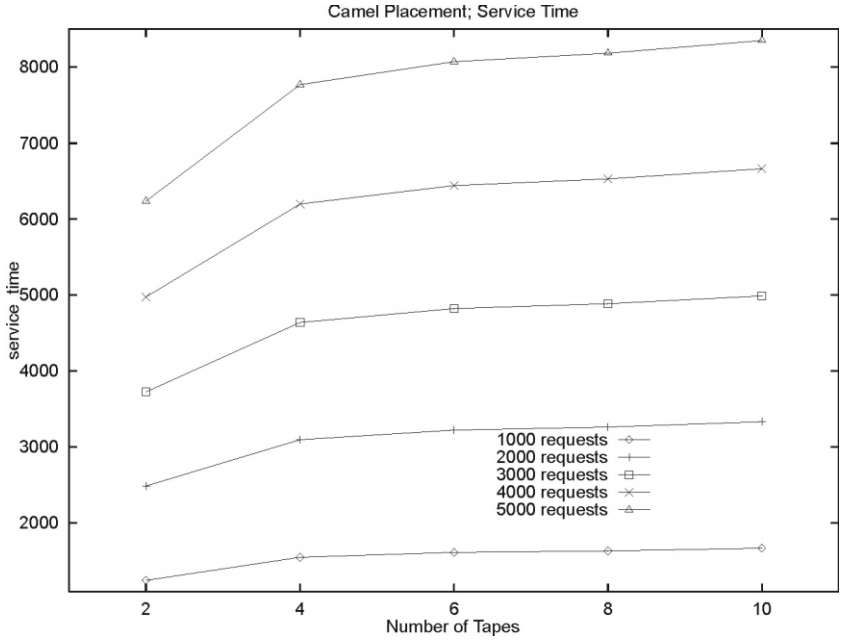

Fig. 13. Camel versus number of tapes.

Figs. 5 and 6 show how the expected values of the chosen performance metrics change for a storage system with a varying number of tapes $(2, \ldots, 10)$. In this case the number of nodes of the

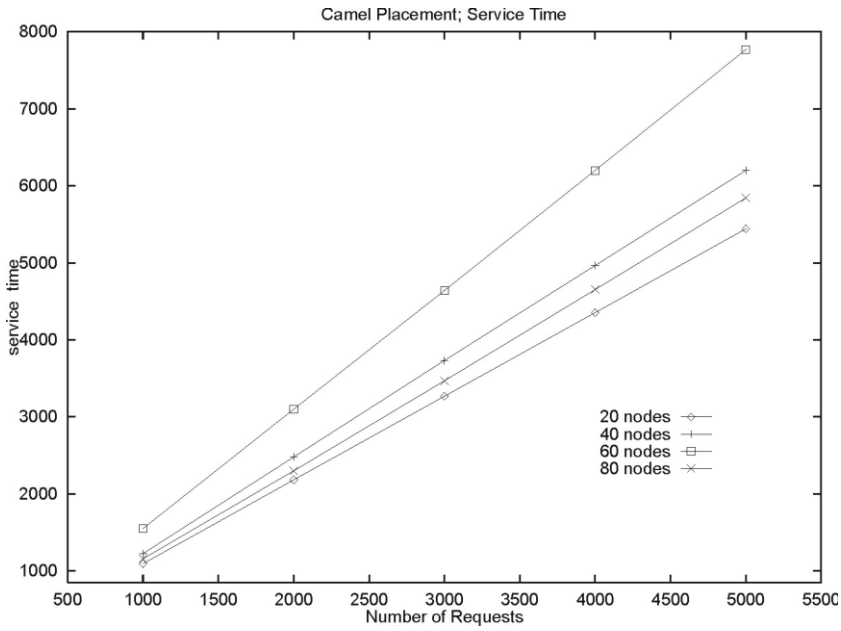

Fig. 14. Camel versus number of nodes.

Table 1
Service–seek time improvement rates between proposed policies and random placement

	Camel		Organ-pipe		Iterative improvement	
	Service time	Seek time	Service time	Seek time	Service time	Seek time
20 nodes	−5%	−7%	15%	19%	20%	27%
40 nodes	1%	1%	21%	27%	31%	38%
60 nodes	−3%	−4%	22%	28%	26%	34%
80 nodes	−1%	−1%	16%	21%	27%	34%

external browsing graph remains constant. No significant variance is indicated by the implementation of the camel placement policy. Organ-pipe placement improves service (seek) time by an average rate of 19% (25%). The experiments indicate that even in this case the implementation of the iterative improvement algorithm, further improves the values of expected service and seek times. In this case, the improvement rates reach 25% and 32% for each one of the performance metrics respectively. A more detailed comparison of the placement algorithms' performance is depicted in Table 2.

6. Future work

A two level (nested-graph) video data representation model has been introduced and based on this model we have adopted certain criteria that guided the placement of data on a tertiary storage system. Experimentation concerning both constructive placement and iterative improvement placement algorithms indicate that iterative improvement considerably improves system's per-

Table 2
Service–seek time improvement rates between proposed policies and random placement

	Camel		Organ-pipe		Iterative improvement	
	Service time	Seek time	Service time	Seek time	Service time	Seek time
2 tapes	–	1%	20%	28%	24%	31%
4 tapes	−3%	−1%	22%	28%	26%	34%
6 tapes	0.1%	1%	18%	23%	26%	37%
8 tapes	1%	2%	17%	22%	25%	31%
10 tapes	1%	1%	17%	22%	27%	33%

formance. Organ pipe placement scheme proves to be better than both camel and random placement. Camel placement has not been proven to be beneficial and it has even resulted in worse seek and service times that random placement. Further research should extend the video data representation models so as to meet the demands of specific video applications, while adopting different criteria to guide video data layout in the overall storage system. Furthermore, information placement algorithms should also be implemented for other types of tertiary and secondary storage systems, including optical and magnetic disks. Moreover, we could extend our model in order to exploit all levels of the storage hierarchy in order to both improve response/service times. For example, we can propose a data placement approach based on objects access frequencies and dependencies, in order to "split" the browsing graph among secondary and tertiary storage levels.

Acknowledgements

This work is supported by the National Science Foundation under Grants 9972883-EIA, 9974255-ISS and 9983249-EIA; grants from HP, IBM, Intel, Telcordia and CERIAS; and Indiana 21st century grant from the State of Indiana.

References

[1] Bertino E, Ferrari E. Temporal synchronization models for multimedia data. IEEE Trans Knowledge Data Engng 1998;10(4):612–31.
[2] Kwon Y-M, Ferrari E, Bertino E. Modeling spatio-temporal constraints for multimedia objects. Knowledge and Data Engng 1999;30:217–38.
[3] Boulos J, Ono K. VOD data management on tape-based storage systems, SPIE Conference on Multimedia Storage and Archiving Systems III, Boston, Massachusetts, November 1998.
[4] Chen Y-T. Physical storage model for interactive multimedia information systems. PhD Thesis, Department of Electrical Engineering, Purdue University, 1993.
[5] Chervenak AL. Tertiary storage – an elevation of new applications. PhD Dissertation, University of California at Berkley, 1994.
[6] Christodoulakis S, Triantafillou P, Zioga F. Principles of optimally placing data in tertiary storage libraries. 23rd International Conference of Very Large Databases (VLDB), Athens, 1997. p. 236–45.
[7] Chung SM. Multimedia information storage and management. Dordrecht: Kluwer Academic Publisher, 1996.
[8] Cox DR, Miller HD. The theory of stochastic processes. London: Chapman and Hall; 1977.
[9] Escobar-Molano ML, Gandeharizadeh S, Ierardi D. An optimal resource scheduler for continuous display of structured video objects. IEEE Trans Knowledge Data Engng 1996:8(3).

[10] Gemmel J, Christodoulakis S. Principles of delay-sensitive multimedia data storage and retrieval. ACM Trans Inform Sys 1992;10(1):51–90.
[11] Ghandeharizadeh S. Stream-based versus structured video objects: issues, solutions, and challenges, Multimedia database systems: Issues and Research Directions, p. 215–36, 1996.
[12] Hirzalla N, Falchuk B, Karmouch A. A temporal model for interactive multimedia scenarios. IEEE Multimedia 1995;2(3):24–31.
[13] Isaacson DL, Madsen RW. Markov chains theory and applications. New York: Wiley; 1976.
[14] Liu JCL, Hsieh J, Du DHC. Performance of a storage system for supporting different video types and qualities. IEEE J Selected Areas Commun 1996;14(7):1314–31.
[15] Prabhakar S, Agrawal D, Abbadi AEl, Singh A. Tertiary storage: current status and future trends. Computer Science Department, University of California, Santa Barbara, 1996.
[16] Sesardi S, Rotem D, Segev A. Optimal arrangements of cartridges in carousel type mass storage systems. The Comput J 1994;37(10):873–87.
[17] Shahabi C, Ghandeharizadeh S. Continuous presentations sharing clips. ACM Multimedia Sys 1995;3(2).
[18] Shahabi C. Scheduling the retrievals of continuous media objects. PhD Thesis, Computer Science, University of Southern California, 1996.
[19] Subrahmanian VS. Principles of multimedia database systems. Morgan Kaufmann Publishers; 1997.
[20] Hua KA, Lang SD, Lee WK. A decomposition-based simulated annealing technique for data clustering. Proceedings of the 13th ACM Symposium on Principles of database systems, Minneapolis, USA, May 1994.
[21] Fleischer M. Simulated annealing: past, present and future. Proceedings of the 1995 Winter Simulation Conference.
[22] Tsao S-L, Huang Y-M, Ding J-W. Performance analysis of video storage server under initial delay bounds. J Sys Arch 2000;46:163–79.
[23] Vakali A, Manolopoulos Y. Information placement policies in tertiary storage systems, storage models for multimedia object. Proceedings of the Hellenic Conference of New Information technologies, October 1998, p. 205–14.
[24] Zimmermann R. Continuous media placement and scheduling in heterogeneous disk storage systems. PhD Thesis, Computer Science, University of Southern California, 1998.

Athena I. Vakali received a BS degree in Mathematics from the Aristotle University of Thessaloniki, Greece and a MS degree in Computer Science from Purdue University, USA. In 1997 she received a PhD degree in Computer Science from the Aristotle University of Thessaloniki. Since 1997, she is a Lecturer at the Department of Informatics, Aristotle University of Thessaloniki, Greece and she has been a visiting professor at Purdue University, USA. Her research interests include design, performance and analysis of storage subsystems and data placement schemes for multimedia and Web based information. She has published several papers in international journals and conferences.

Evimaria Terzi graduated from Eniaio Polykladiko Lyceum of Katerini, Greece. She has succeeded the first grade at the national exams for the Informatics Department at the Aristotle University. She has received her BS degree in Computer Science from the Aristotle University of Thessaloniki and she has been granted a research assistantship for graduate studies at Purdue University, USA. She has received the Fulbright scholarship for graduate studies in the US and she has published papers on tertiary storage systems. Her research interests include storage subsystem's performance multimedia databases and data placement schemes.