Mapping Media Streams onto a Network of Servers*

Reinhard Lüling

Department of Mathematics and Computer Science
University of Paderborn, Germany
rl@uni-paderborn.de

Abstract

This paper presents the definition as well as a number of methods for the solution of a new combinatorial optimization problem, called S-MAMP that has to be solved for the efficient management of a network of media servers. This network of servers can be implemented on top of a closely connected network of workstations as well as on a wide area network of media servers. The problem studied here addresses the management of media assets on such a network, i.e. the placement/replication of media assets on a net- work of media servers taking the user requests, storage capacity and bandwidth limitations of the server network into account. The algorithms presented for the solution of the S-MAMP problem make efficient use of the ability of audio/video encoding standards such as MPEG to encode one stream in different bitrates providing scalable service quality to the user in this way.

1 Introduction

1.1 Problem motivation

The mapping problem is one of the key problems in parallel and distributed computing. A large number of results have been published in the past on the problem of mapping data structures and communicating processes onto a network of processors either building up a massively parallel computer system, a network of closely connected workstations or a widely distributed computing or information system. Most of the mapping problems studied in the past perform a mapping of computational intensive processes or datastructures onto a network of processors to distribute the processing workload as evenly as possible in order to speed up the overall computation time using the resources of the parallel or distributed system in the most efficient way.

In this paper we investigate a new mapping problem that arises when distributed information systems are build up. We study the implementation of a scalable server system for the delivery of continuous media, e.g. audio and video to a large number of clients. This problem has gained considerable attention, as the storage capacity as well as the communication throughput of modern infrastructures allow to store and deliver audio/video streams via Intranet / Internet environments to client systems (PCs or TV sets) in a very good quality.

Approaches to build up large scale video servers currently followed in first industrial applications as well as in the literature focus on the implementation of such systems on top of parallel computing systems. These type of video servers usually connect a large number of storage devices, e.g. disks and memory modules to a large number of communication devices to deliver the audio/video streams from the storage devices to the user clients. Some examples for the architecture of media servers build up on top of parallel computing systems can be found in [2].

In this paper we follow an alternative approach for the implementation of scalable media server systems: Instead of building up a single large video server that usually results in a asymmetric network structure, we connect a number of small video servers by a backbone network resulting in a distributed server network. Each of the nodes of this network now connects to a number of client systems. A network of media servers can be used to implement an information system build up by a network of workstations within a local Intranet, but it can also be used to implement a widely distributed information systems using large scale video servers as nodes in a wide area communication network. Figure 1 presents a network of media servers and their local clients.

A mapping problem now arises when a set of media assets (audio/video media streams encoded according to some standard, e.g. MPEG) have to be mapped onto the server network. It is obvious, that a client being connected to one

*This work was partly supported by the MWF Project "Die Virtuelle Wissensfabrik", the EU project SICMA, the EU project EPRI-COM and the BMBF project HiQoS.

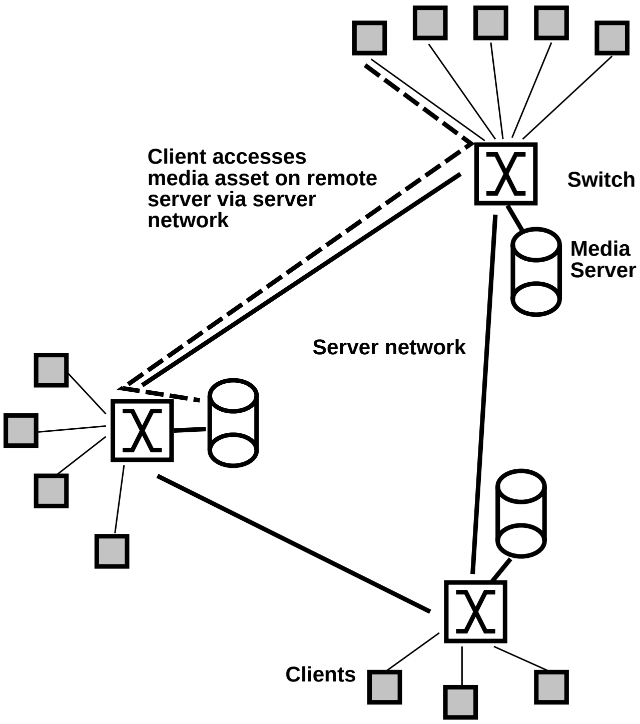

Figure 1. Distributed media servers

of the servers has easy (and cheap) access to all assets stored on its server, but the access to media assets stored on a remote server is more costly as the backbone network has to be used to stream the media asset from the hosting server via the local server of the client to the client itself. One should notice, that for media streaming the communication lines are not only used to transfer the media asset in one large packet, but a continuous stream of media information is set up that delivers small packets according to the bitrate of the stream continuously until the client ends the viewing session of the selected stream. In this way, a load which is determined by the bitrate of the stream (e.g. 1.5 MBits/sec for MPEG-1 encoded audio/video streams) is put on the path from the hosting server to the client in the overall network.

If each client is now allowed to access a number of media assets provided by the information service that is implemented on the server network, the questions comes up on what servers these media assets should be stored to allow all clients to access their media information (notice that copies can be easily managed as the data items which are the media assets in this case are read-only). To solve this mapping problem, tradeoffs have to be studied regarding the communication bandwidth of the backbone connecting the servers, the storage capacity of the servers as well as the bitrate of the audio/video streams provided by the information service. The nice thing with audio/video streams is, that a single stream can be provided in various bitrates. Thus, if a media asset is provided by an information service the bitrate can be chosen according to the quality the end user is expected to get but also according to the available resources in terms of storage capacity and communication bandwidth. That is, the overall information of a media asset being encoded in a low bitrate is the same as encoded in a higher bitrate but the presentation is not as nice, because the audio/video stream may be encoded with less pictures per second or with a smaller window format or by decreasing other encoding parameters. Thus, the tradeoff that has to be solved by the mapping problem is to provide to each client the selected media assets in the highest possible quality (maximal bitrate) taking the storage capacity of the server network as well as the communication capacity of the network into account. In this way the 'Quality of Service' (QoS) can be optimized taking the restrictions of the network of servers into account. Thus, the placement of media streams on a network of servers defines a new mapping problem. The most important new feature here is, that the size of the media assets (determined by the bitrate used for encoding) is not fixed but can be optimized taking the restrictions of the underlying hardware into account.

This problem has a large number of possible variations. In the motivation given above, we took the storage capacity and the communication network as a given parameter and optimized the quality of service, i.e. the bitrate of the audio/video streams. Regarding the planning of a network of media servers, one could also ask what is the minimal storage capacity and bandwidth of the communication network to provide a number of media assets in a given quality. A large number of other questions are also possible. Thus, this problem has large variations for the set-up of networks of media servers either as a local workstation cluster or a wide area distributed information system.

1.2 Previous Work

Within this paper we study the question of optimizing the QoS taking the restrictions of the server network into account. This problem has some relation to the 'File Allocation problem (FAP)' studied in a large number of papers (see [5] for an overview of the most important models). The FAP is a classical optimization problem but always considers the files that have to be placed onto a network as a fixed parameter without being able to change its size as it is the case for media streams handled here. In this way, the problem studied here extends the work on the FAP. Similar mapping problems as studied for the FAP have also been studied for the problem of assigning global variables to a network of processors in a MPP system. As the data items placed here are much smaller as the files in the FAP or the media assets to be mapped in the problem studied in this paper completely other strategies are applied there.

The problem studied in this paper has also been described in [10] and [11]. In these two papers presented by Venkatasubramanian and Ramanathan no solutions for the mapping problem were discussed, but some general principles for duplicating media assets on a network of servers were presented. The methods presented in this paper therefore extend the work of Venkatasubramanian and Ramanathan as our algorithms compute an optimized mapping of media assets and do not rely on some general principles how to copy media assets on such a network as done in the previous papers.

1.3 Results presented in this paper

In this paper we describe a complete model (in section 2) of the problem motivated above, called the 'Static Media Asset Mapping Problem (S-MAMP)'. We present several methods to solve this optimization problem. The main contribution of this paper is the presentation of heuristics (given in section three) for the solution of the S-MAMP problem. Based on the well known principle of local search we identify a number of neighborhood structures for the definition of local algorithms. Using these algorithmic components, we study a number of simulated annealing algorithms composed out of these basic components. The performance of the presented methods is studied (in section four) using a number of benchmark instances. It is shown that the neighborhood structure of the local search algorithms largely effects the performance of the algorithms and we identify some algorithms that solve the presented optimization problem close to optimality respectively close to a given upper bound. In this way the paper presents a number of methods for the efficient solution of the S-MAMP problem.

2 Definition of the S-MAMP problem

In this section we present a formal model of the S-MAMP problem. In fact we will model the network of media servers first, then give a definition of the access pattern to the media assets and then define the optimization problem that has to be solved to map the media assets onto the server network.

2.1 Model of network of media servers

A network of media servers can be modeled as a graph $N = (H, E)$, $|H| = n$, of n servers connected by a number of communication lines. Each server (host) has a given capacity that can be used to store media assets. This is represented by a capacity function $c : H \rightarrow I\!N$ that assigns an integer number to each host, representing the MByte of storage capacity of the media server. Each edge $e \in E$ represents a bidirectional communication line connecting two servers in the server network. The function $w_e : E \rightarrow I\!N$ assigns the communication capacity in MBit/sec to each communication line. We assume that the network connecting the user clients to the media servers provides suitable bandwidth to retrieve media streams from the server.

The network of media servers stores a set of media assets $M = \{a_1, \ldots, a_m\}$. The access of user clients (being connected to the media servers) to the media assets is modeled as a graph $A = (V_A, E_A)$ and $V_A = H \cup M$, $E_A \subseteq \{\{v, a\} \mid v \in V,\ a \in M\}$. Thus, an edge $(v, a) \in E_A$ indicates that some client connected to server v aims at receiving asset a from the network of media servers.

Media assets can now be provided in different qualities by the server network to the users. This is possible by encoding the media assets in different bitrates (MBit/sec). The set of suitable bitrates is given by a set $B \subseteq I\!N$.

2.2 Definition of problem variations

For a given network of servers N, an access pattern A and a set of bandwidths B the question is now whether it is feasible to map the media assets in such a way onto the media servers that each access request can be fulfilled. An access request is fulfilled if a host u requesting a media asset a stores this asset directly or a path in the communication network to another host v hosting media asset a provides sufficient bandwidth to stream the asset to the requesting host u.

To fulfill the access requests a number of "tools" are available that can be used: Media assets can be mapped redundantly to different hosts, the encoding bandwidth (and therefore the space requirement) of media assets can be scaled, and the communication lines can be used to stream media assets via the network to remote clients. Using these tools, we can formalize the following decision problem:

given: A server network $N = (H, E)$, an access structure $A = (V_A, E_A)$ and a set possible encoding bitrates B

Question: Is there a mapping $\pi_R : A \rightarrow P(E)$ and $\pi_b : A \rightarrow B$ with $\pi_R((v, a)) = \{(v, v_1), (v_1, v_2), \ldots, (v_{l-1}, v_l)\}$ for an $l \in I\!N_0$ and $\pi((v, a)) = v_l$ and $\pi_b((v, a)) = b$ such that :

$$\sum_{(v,a)\in E_A,\, \pi((v,a))=h} \pi_b((v, a)) \leq c(h)\ \forall h \in H \text{ and}$$

$$\sum_{\substack{(v,a)\in A,\, (u_1,u_2)\in \pi_R((v,a)) \\ e=(u_1,u_2)}} \pi_b((v, a)) \leq w_e(e)\ \forall e \in E,$$

Based on this problem definition we can now formulate the optimization problem. This problem addresses the tradeoff among the storage capacity of the media servers, the communication bandwidth of the network and the quality (bitrate) in which the media assets can be retrieved by the clients. The optimization problem is formalized as follows:

given: A server network $N = (H, E)$, access structure $A = (V_A, E_A)$, a set of possible encoding bitrates B

Optimization: Find a mapping $\pi_R : A \to P(E)$ and $\pi_b : A \to B$ with $\pi_R((v,a)) = \{(v, v_1), (v_1, v_2), \ldots, (v_{l-1}, v_l)\}$ for an $l \in I\!N_0$ and $\pi((v,a)) = v_l$ and $\pi_b((v,a)) = b$.

$$\sum_{(v,a)\in E_A, \pi((v,a))=h} \pi_b((v,a)) \leq c(h) \; \forall h \in H \text{ and}$$

$$\sum_{(v,a)\in A, (u_1,u_2)\in \pi_R((v,a))} \pi_b((v,a)) \leq w_e(e) \; \forall e \in E,$$

$e = (u_1, u_2)$

and $QoS := \sum_{(v,a)\in E_A} \pi_b((v,a)) \to \max$

In the same way as we ask for a maximal bitrate of each requested stream we can also ask for a minimal storage capacity of the servers or a minimal bandwidth of the communication lines. Thus, the optimization problem can be formulated with different objectives.

3 Algorithms for the S-MAMP problem

Local search algorithms have been shown to be very efficient for the solution of combinatorial optimization problems. The local search principle has been used for classical graph theoretic problems, e.g. the Traveling Salesperson Problem and the graph partitioning problem, but also for very specific problems that have a large number of restrictions and arise in industrial applications. A local search algorithm defines a neighborhood relation (graph structure) on the set of all feasible solutions of a given optimization problem. Starting from a randomly choosen feasible solution, a search process is performed on the network of feasible solutions, identifying one node of this graph as the current solution in each iteration. The various local search algorithms can be differentiated by the way a new current solution is accepted.

In case of a "hillclimbing algorithm" a neighboring node is always accepted as the new solution if the value of the cost function is improved (we consider maximization problems here). The algorithm terminates if in a number of iterations the cost function value could not be improved. In the case of the simulated annealing algorithm the overall process is controlled by a parameter t, called the temperature of the annealing process. In each iteration the temperature is decreased by a given factor. The process is terminated if the parameter t reaches a given final temperature. A new solution is always accepted if it improves to cost function value. It is also accepted in case of the lower cost function value with a probability $e^{\frac{-\Delta}{t}}$, if Δ is the difference of the new cost function value and the old one. For the simulated annealing principle it was shown that the probability that an optimal solution is found converges to 1 if the number of iterations goes to infinite.

Within this section we present a number of algorithms for the solution of the S-MAMP problem. To do this, we first present a number of algorithmic components that are later on used for the construction of the optimization algorithms.

3.1 Neighborhood structure - Phase I

For a given configuration s being a feasible solution of the S-MAMP problem the computation of a neighboring configuration s' is done in two phases. In a first phase, a node in the network of servers is identified and the encoding bitrate of one or more media assets mapped onto this server is increased respectively decreased. This pertubation of the current solution also effects other hosts which access the media asset from the selected server. Thus, in the second phase of the computation of a local neighbor the connection of non-local media assets is re-assigned.

We present three different algorithms for the pertubation of media assets being stored on a single server. The first algorithm is rather simple as it selects a random host and media asset and increases the bitrate of the asset to the next available bitrate. After this, random assets on this host are choosen to decrease their bitrates until the overall capacity restriction is met again. The algorithm is listed in Figure 2

The second algorithm also chooses a random host first. An asset is choosen at random from those media assets that are accessed by the choosen host. The bitrate of this asset is increased in a larger step compared to the first algorithm. To met the capacity restrictions assets stored on the server are choosen randomly and their bitrates are decreased. In this algorithm the assets that decrease their bitrates are choosen with exponential distribution giving a higher priority to those assets who are mapped with small bitrates only onto the server. In this way redundant copies which only have smaller bitrates are deleted with higher probability.

The third algorithm (algorithm I.3) used for the first phase follows the scheme of the second one, but takes the number of hosts into account that access a media asset stored on the local server and not the bitrate of the media asset when selecting an asset for decreasing its bitrate. In this way, media assets that are accessed by a large number of other remote servers are receiving a decrease of their bitrate or a total deletion with a very small probability only.

a node $h \in \{1, \ldots, n\}$ and asset $a \in M$ are choosen at random (uniform distribution)
if a is mapped onto h the bitrate is increased to the next possible bitrate
if a is not mapped onto h then map it with the smallest possible bitrate onto h
while (current load of host h exceeds $c[h]$)
    an asset a is choosen randomly with uniform distribution
    if a is mapped onto h and a encoded with the lowest bitrate then delete a from h
    if a is mapped onto h decrease the bitrate of a to the next possible bitrate

Figure 2. First phase of neighborhood computation

3.2 Neighborhood structure - Phase II

In the second phase of the algorithm for the computation of a neighboring configuration all edges (v, a) of the access pattern for a host $v \in H$ and media asset $a \in M$ where a is not available on host v are considered. These edges have to be mapped onto the network, i.e. the media stream has to be routed from a host storing a to v. We define the following algorithm presented in Figure 3 for this:

3.3 Computing an initial solution

For a given instance of the S-MAMP problem we have seen above that already the decision problem is NP-complete. Thus, to find an initial solution for a given instance is already an NP-complete problem. We assume therefore in the following, that the storage capacity as well as the network capacity is sufficient to allow for a random mapping of each media asset onto the network of servers. A very simple algorithm places each media asset usin the smallest possible bitrate on exactly one host. Then it uses the algorithms for the first phase to map each edge of the access graph onto the server network.

The second algorithm uses the same scheme but increases the bitrate of each asset that is assigned to its maximum and maps additional copies of media assets on hosts that request this asset and still have available storage capacity.

3.4 A simulated annealing algorithm

Using the two algorithms for the computation of an initial solution, the three algorithms for the first phase of the neighborhood computation and the algorithm for the second phase, a number of 6 different algorithms can be constructed using the given basic building blocks.

4 Performance evaluation

In this section the performance of the algorithms presented in the previous section will be investigated in detail. To do this, we define a set of benchmarks instances that reflect the implementation of large scale distributed video archives and their typical access pattern. But before we do this an upper bound for the solution quality is given that is used later on to evaluate the quality of the solutions computed by the algorithms.

4.1 Upper bound for the solution quality

The achievable quality that can be reached by one host in the network is bounded by the hardware provided to map a sets of media assets on, i.e. the capacity of the hosts and that of its associated communication lines.

If the provided communication bandwidth of the network or the capacity of the hosts cannot be completely used by the access pattern, the quality that can be reached by one host is bounded by the number of requested media assets of this host.

All over all, the following upper bound is valid for a given network, access pattern and bandwidth:

$$\sum_{i=1}^{n} \min\{c_i + \sum_{j=1, j\neq i}^{n} w_e((j,i)), \max_{b \in B} b \cdot \mid \{(i,j) \in E_A\} \mid\}$$

4.2 Definition of benchmark set

The benchmark instances assumes a clique network for the connection of the host as well as a constant server capacity and bandwidth of the communication edges. Thus, a benchmark instance is defined by the number n of hosts, the capacity c of each host and the communication bandwidth w_e of each communication line, the number of media assets m and the access pattern. We assume that each asset can be encoded using a number of bandwidths given by the set B.

Table 1 presents the details for the different benchmark classes. All access patterns were randomly generated. The first set of benchmark instances (marked R) are built using access patterns where a host chooses an asset with a given uniform distribution. The second class of access patterns (marked E) use an exponential distribution for the access of media assets. These patterns reflects the case for the implementation of a video library that contains newer and older movies. These kind of access patterns are typical for digital librarires containing Hollywood movies.

```
for h := 1 to n do
    for j := 1 to n do bw_in(j) := 0
    for all a ∈ M with (h, a) ∈ A and a is not mapped onto h
    let K be the set of all hosts that store a copy of asset a
    while not successful and K ≠ ∅ do
        choose a host h' ∈ K with bitrate of asset a on h' is maximal
        K := K − {h'}
        if bw_in(h')+ bitrate of a on host h' ≤ w_e((h', h)) then
            successful:=true; asset a is streamed from host h' to h
    if not successful then 'access to asset a from host h not possible'
```

Figure 3. Second phase of neighborhood computation

Benchmark Class	n	m	c	w_e	B	access pattern
R_32_256_750_100_0.5	32	256	750	100	$\{5, 10, 15, 20, 50\}$	random $p = \frac{1}{2}$
R_32_256_1000_100_0.5			1000			
R_32_256_1500_100_0.5			1500			
E_32_256_750_100_0.5	32	256	250	50	$\{5, 10, 15, 20, 50\}$	exponential

Table 1. Definition of benchmark instances using a clique network

4.3 Performance results and discussion

In the following the performance of the various algorithms and benchmark instances is discussed. Table 2 shows the resulting QoS computed by the simulated annealing algorithms as well as the upper bound for the benchmark instances defined above. In detail the table shows the results for the 6 different benchmark classes and 6 algorithms and compare the performance to the upper bound (difference in percent). For each benchmark classes 5 instances were generated that were run 10 times using the (randomized) simulated annealing algorithms. The values presented in the table are the mean values of the solutions (QoS) found by the 10 runs of the algorithms.

The measurement results show, that the difference between the upper bound and the results gained by the simulated annealing algorithm are very small. They range from about 10 percent down to less than 3 percent. As this is the comparison to an upper bound and not to the optimal solution, it can be concluded that the algorithms are able to find very good heuristical solutions. Comparing the different methods presented in the previous section, it can be concluded that the more intelligent neighborhoods lead to better results than the simple algorithm for the first phase. A comparison of the algorithms for the computation of the initial solution shows that the second algorithm provides better solution quality here.

5 Conclusion and Future Work

This paper studies the efficient implementation of large scale distributed video on demand systems using a network of regionally distributed video servers. One of the most important problems for the implementation of server networks is the assignment of media assets onto the server network. We defined the S-MAMP (Static Media Asset Mapping problem) as a new combinatorial optimization problem that models the different trade-offs that have to be considered when designing and operating such a network of media servers.

To solve the specified optimization problems we present different heuristics for the solution of the problem studied here. Using a set of benchmark problems it can be seen that the defined problem can be solved efficiently achieving near to optimal solutions in short time. Using parallel algorithms of simulated annealing, the solution time of the presented simulated annealing algorithms can be further decreased. In the optimal case, this parallel algorithm can be performed on the network of servers using the computational power that is available there.

The relevance of the problem and the algorithms for its solution presented here are motivated by the development of a large scale distributed VoD system for the implementation of a European information systems connecting parliaments of central and eastern Europe to the European parliament in Brussels. Thus, the algorithms presented here will be used in a real world scenario.

The algorithms studied within this paper determine a static mapping of a set of media assets to a network of media servers optimizing the trade-off between server capacity communication bandwidth of the network and the quality with which the audio/video streams are provided. The algorithm uses a static access pattern for the determination of the mapping. If the access pattern changes dynamically, which

	SA Algorithms						Bound
Initialization	1	1	1	2	2	2	
Phase I	1	2	3	1	2	3	
R_32_256_750_100_0.5	108690	112553	113425	109772	116980	118224	123200
	13.34 %	9.45 %	8.61 %	12.23 %	5.31 %	4.20 %	
R_32_256_1000_100_0.5	118265	122540	123348	119329	124992	124873	131200
	10.93 %	7.06 %	6.36 %	9.95 %	4.97 %	5.07 %	
R_32_256_1500_100_0.5	132883	140195	140856	134331	141552	140565	147200
	10.77 %	4.99 %	4.50 %	9.58 %	3.99 %	4.72 %	
E_32_256_250_100_0.5	12390	12563	12433	12690	12930	12754	13300
	7.34 %	5.86 %	6.97 %	4.81 %	2.861 %	4.28 %	

Table 2. Performance Results

is usually the case in practical applications, a remapping has to be performed to optimize the overall cost function. Our future work in this area will therefore study these dynamic access patterns, specifying the 'Dynamic Media Asset Mapping Problem' (D-MAMP), and the methods that have to be applied for this case.

References

[1] W. C. Chiang, R. A. Russel: *Simulated annealing metaheuristics for the vehicle routing problem with time windows*, Annals of Operations Research, vol. 63, 1996, pp. 3-27.

[2] F. Cortes Gomez, R. Lüling: *A Parallel Continuous Media Server for Internet Environments*, Proc of Int. Conf. on High-Performance Computing and Networking (HPCN Europe'98), Lecture Notes in Computer Science, pp. 78-86

[3] G. Dammicco, U. Mocci: *Program Caching and Multicasting Techniques in VoD Networks*, Proc. of 4th International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services (IDMS), 1997, pp. 65-76

[4] R. Diekmann, R. Lüling, J. Simon: *A General Purpose Distributed Implementation of Simulated Annealing*, Proc. of the 4th IEEE Symposium on Parallel and Distributed Processing (SPDP'92), 1992, pp. 94-101

[5] L. W. Dowdy, D. V. Foster: *Comparative Models of the File Assignment Problem*, Computing Surveys, vol. 14, no. 2, 1982

[6] A. Leff, J. L. Wolf, P. S. Yu: *Replication Algorithms in a Remote Caching Architecture* , IEEE Transactions on Parallel and Distributed Systems, vol. 4, no. 11, November 1993, pp. 1185 - 1204.

[7] D. Ma, G. Alonso: *Client Caching Algorithms in a Video-on-Demand System*, Technical Report, ESPRIT LTR project HERMES, Department of Computer Science, ETH Zürich.

[8] B. Monien, I.H. Sudborough, *Embedding one Interconnection Network in Another*, Computing Suppl. 7, 1990, pp. 257-282.

[9] J. P. Nussbaumer, B. V. Patel, F. Schaffa, J. P. g. Sterbenz: *Networking requirements for Interactive Video on Demand* , IEEE Journal on Selected Areas in Communications, vol. 13, no. 5, June 1995.

[10] N. Venkatasubramanian, S. Ramanathan: *Effective Load Management for Scaleable Video Servers* , HP Laboratories, Technical Report, March 1996, and on the WWW at http://yangtze.cs.uiuc.edu/˜nvenkata/

[11] N. Venkatasubramanian, S. Ramanathan: *Load Management in Distributed Video Servers*, Proc. of Int. Conf. on Distributed Computing Systems (ICDCS), 1997, pp. 31-39