Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 03-036

Proxy-Assisted Periodic Broadcast for Video Streaming with Multiple Servers

Ewa Kusmierek and David Du

September 17, 2003

Proxy-Assisted Periodic Broadcast for Video Streaming with Multiple Servers

Ewa Kusmierek and David H.C. Du
Digital Technology Center and
Department of Computer Science and Engineering
University of Minnesota
{kusmiere, du}@cs.umn.edu

***Abstract*—Large scale video streaming over Internet requires a large amount of resources such as server I/O bandwidth, network bandwidth. A number of video delivery techniques can be used to lower these requirements. Periodic broadcast by a central server combined with proxy caching offers a significant reduction of the aggregate network and server I/O bandwidth usage. However, the resources available to a single server are still limited. In this paper we propose a system with *multiple geographically distributed servers*. Multiple servers beside offering increased resources and service availability, allow a further reduction of network bandwidth usage. The challenge is how to use multiple servers efficiently. We first analyze the dependence of the resource requirements on the number and locations of the servers in a proxy-assisted periodic broadcast video delivery system. Based on the character of the function describing such a dependence, we formulate and solve the problem of video location and delivery in a way that minimizes resource usage. We explore the trade-offs between network and I/O bandwidth requirements. We evaluate our proposed solutions through a number of tests.**

Keywords: video streaming, periodic broadcast, proxy caching, distributed servers.

I. Introduction

Many multimedia applications such as distance learning and video-conferencing rely on video streaming techniques. However, large scale video delivery requires a large amount of resources such as storage space, server I/O bandwidth and network bandwidth. A number of techniques have been proposed to reduce these requirement and to address the scalability of video delivery system. Periodic broadcast [1] reduces server I/O bandwidth and network bandwidth requirements for popular videos by segmenting a video and repeatedly broadcasting these video segments over a fixed number of channels. The network bandwidth requirement reduction can be also achieved by employing proxy servers to cache data [2] at close proximity to the clients. Proxy servers strategically placed at the boundaries between Wan Area Network (WAN) and Local Area Network (LAN) utilize their processing and buffering capabilities.

The two approaches are combined into a proxy-assisted periodic broadcast [3, 4]. By caching partial video content at a proxy and broadcasting the remaining content at a central server we can achieve a significant resource requirement usage reduction. However, in a large scale of the video delivery system, these requirements can still be considerably large. Even if the resource requirements are independent of the number of clients accessing each popular video as in periodic broadcast schemes, the total requirements still depend on the number of available videos. Proxy servers have also limited capabilities, hence, their ability to save the required network bandwidth is also limited.

In this paper we propose to employ multiple video servers to further reduce the resource requirement. Multiple server system offers three major benefits: 1) an increase in the amount of resources available such as I/O bandwidth and storage space, 2) increased service availability and fault tolerance due to replication, and 3) further reduction of resource requirement such as WAN bandwidth by allowing clients to access video from a set of geographically distributed servers. A server cluster can be used to increase resource availability, while reduction of resource usage requires a geographically distributed system. Since we consider video streaming over the Internet, network bandwidth is an important factor and hence we concentrate on the geographically distributed server system.

The challenge in efficient utilization of multiple server system lies in how to characterize the dependence of the aggregate I/O bandwidth and WAN bandwidth requirements on the number and locations of servers employed. WAN bandwidth usage can be intuitively reduced be deploying multiple distributed servers due to the reduction of distance between clients and servers. I/O bandwidth on the other hand, exhibits different and more complex behavior. We first show that the minimum I/O bandwidth usage is obtained with a small number of servers and how the distribution of client requests impacts I/O bandwidth usage. We then formulate the problem for distributing videos and client requests for these videos among a number of potential servers that allows to reach any desirable level of I/O bandwidth usage including the minimum value. We also explore the trade-offs between I/O bandwidth and WAN bandwidth requirements.

We consider settings with a number of video servers available (Figure 1). The clients are grouped into communities according to their locations, and are generally connected to the same LAN, the same cable head-end or the same Internet Service Provider. Communities are spread geographically over the Internet. Each community has a proxy server providing caching service. The client requests that cannot be satisfied by the proxy are directed to one of the geographically distributed servers. Servers have more resources available than proxies, but there are fewer

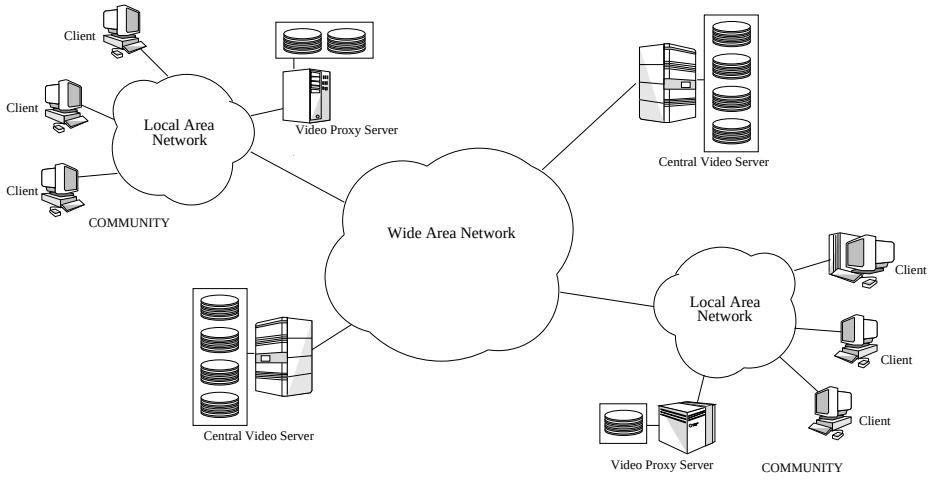


Fig. 1. Video delivery architecture

servers than communities. Hence, each server provides service to a number of communities. The problem we address is *video placement*, i.e., deciding which servers each video is available from, and *video delivery*, i.e., distributing client requests for a video among a number of available servers. The goal is to minimize the resource requirements while maximizing the number of accepted requests. We consider the uncapacitated version of the problem for which minimizing the resource consumption is the main target, as well as the capacitated version, in which each server is assumed to have limited capacity. We explore the trade-offs involved in video replication among a number of servers, namely the influence the replication has on the I/O bandwidth and network bandwidth requirements. Based on the findings we propose a schemes for video location and delivery.

We examine also a different aspect of a multiple server system. That is, given a number of possible server locations, what is the best capacity assignment to each location? We use the model of the streaming video workload to determine what capacity assignment provides the best basis for video location and delivery in the capacitated version of the problem.

The paper is organized as follows. In Section II we summarize related work on the proxy assisted periodic broadcast and some graph theoretic approaches to object (video) placement and delivery problem. In Section III we present the results of the analysis of resource dependence on the number and locations of servers. We formulate the optimization problem for video placement and delivery in Section IV. In Section V we present heuristic solutions to the optimization problems. The test results and evaluation of the heuristics are given in Section VI. We conclude the paper in Section VII.

II. Related Work

The problem of object location and object delivery in a distributed server system has been considered for the web and multimedia content. Some of the aspects of the solutions proposed for the web content delivery are applicable to the video on-demand systems. However, certain characteristics of the multimedia objects such as high I/O bandwidth requirements and duration of the transmission, pose additional challenges. Our approach to video delivery in a distributed multi-server system is based on the framework combining proxy caching with periodic broadcast, and as such differs from other distributed multimedia systems [5–7].

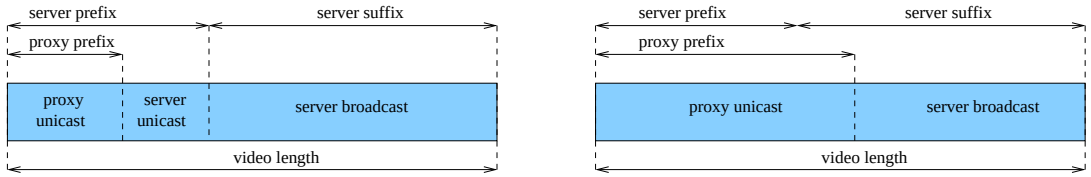
In this section we first describe the general framework for video delivery involving central servers, proxy servers and clients. Next we compare the requirements of such a system with the requirements of the web content delivery and point out how the experiences gained with the web system can be utilized in multimedia system.

A. Proxy-Assisted Periodic Broadcast

A number of schemes have been proposed to address video streaming resource requirements at the server. Bandwidth reduction is achieved by exploiting data sharing among clients. Batching and patching [8–11] group clients to share one transmission by delaying earlier requests, and allowing a client to join-on an on-going transmission while receiving the missed video part on a separate channel.

PB schemes [1,12–14] partition a video into a number of segments. All segments are repeatedly accessed and transmitted by a central server over a number of broadcast channels. Each client collects video segments from these channels and buffers them until their playback times. The total bandwidth required for broadcast transmission does not depend on the number of clients, but only on the segmentation of the video, the number of channels and the transmission rate of each channel. Thus, the scalability is improved for accessing popular videos. For in-frequently accessed videos, unicast of individual stream is still more efficient. Each client has to collect all video segments, therefore, the network bandwidth requirement is not reduced unless some sharing of transmission of video segments over network with other clients is possible. Catching [15] and MPatch [16] combine PB with patching and batching. Selective catching [15] explores potential reduction dependent on the video popularity, i.e., the guidelines are provided for which method, PB or patching and batching, results in higher reduction.

Another approach to resource requirement reduction is based on video caching by a proxy [2, 17]. A proxy caches a part of a video so that the amount of data that has to be delivered by a



(a) proxy prefix $<$ server prefix

(b) proxy prefix $\geq$ server prefix

Fig. 2. Video transmission modes

central server is decreased. Part of the video cached by a proxy is obtained by a client from the proxy in a given community, and the remaining part is delivered by a central server. Among a number of ways to decide which part of the video should be cached [18, 19], *prefix caching* offers an additional advantage of reducing start-up delay. Client receives video prefix from the proxy and starts playback immediately. In the meantime, the client starts also reception and buffering of video suffix from a central server. Proxy caching has been combined with PB in [3, 4] in-to a proxy-assisted periodic broadcast.

All of the above-mentioned schemes target minimization of the resource requirement under the assumption that the video set is provided by *one central server*. In our approach we take one of the techniques, proxy-assisted PB, one step further to explore the possible resource reduction with *multiple servers*. In the rest of this section we describe proxy-assisted PB in more details. First we present a simple scenario of cooperation between a single proxy (or a set of homogeneous proxies) and a single central server introduced in [3]. Next we present a system with multiple heterogeneous proxies and a central server introduced in [4]. In the third step we extend the system to include *multiple heterogeneous servers* with heterogeneous proxies, which is the subject of this paper.

a) One Proxy - One Server: In a system consisting of a single proxy and single central server, a set of *popular* videos is considered. Due to high video access popularity, each video is partitioned into two parts: a prefix and a suffix. The central server uses a PB scheme as an access and transmission mode for the suffix of the video. Proxy caches some prefix of the video in order to reduce the number of channels broadcasted by the servers, each video segment cached by the proxy reduces the number of channels by one. Since the storage space and I/O bandwidth at the proxy are limited, the available space is optimally partitioned among videos in such a way that the number of channels broadcasted by the server is minimized. Hence, the system achieves the minimization of the aggregate server I/O and network bandwidth requirements.

b) Multiple Proxies - One Server: In a system with multiple proxies, we assume that the proxies differ from each other with respect to the storage space and I/O bandwidth available at each proxy. In addition to proxy heterogeneity, also video popularity varies from one community to another and the video set contains videos of *various* popularities. A proxy chooses the prefix for each video in such a way that the total amount of data that has to be transferred by the central server is minimized, subject to the I/O bandwidth and buffer size constraints at the proxy. However, a proxy partitions the video based on its local video popularity and its own capacity.

A central server can deliver video in one of two modes: either through unicast or through periodic broadcast. The choice between unicast and PB transmissions is based on the global popularity of a video, i.e., the aggregate popularity determined by the access frequency from clients in all communities. The global popularity has to be high enough for PB to reduce the resource requirements as compared to unicast transmission.

Due to the fact that different proxies may cache different prefixes of a video, global popularity of a video segment (note not the whole video) at the server is not fixed for the whole video and may vary for different segments. For examples, the initial few segments of a video which are included in each proxy prefix have a global popularity equal to 0. This is due to the fact that no client needs to access these video segments from the server. The global popularity of a video segment reflects the fact that some of the proxies already cache the segment and clients do not need to access that segment from the server.

Based on the global popularity of a video segment, in order to achieve maximum resource usage reduction we allow the choice between unicast and PB transmission to be made for different parts of the same video. Since the global popularity increases as we move from the beginning toward the end of the video, there is a cut-off point that determines the division of the video into a *server prefix* delivered through unicast transmission and *server suffix* delivered through broadcast transmission. Note that if the server prefix is smaller than a proxy prefix, the required data is already cached at the proxy and no data needs to be delivered by the server. If the server prefix is larger than the proxy prefix, only the difference needs to be delivered by the server via unicast. The suffix portion of a video is periodically broadcasted to all clients. For a not-so-popular video, it is possible the server prefix is the whole video and the proxy prefix is 0. Therefore, the whole video is unicasted by the server if there is a request for such a video. A number of combinations of video delivery modes are possible (Figure 2), from having the whole video cached at the proxy to the whole video delivered via periodic broadcast transmission or unicast by the central server.

The granularity of the sizes of both proxy prefix and server prefix is limited by the video segmentation devised for the periodic broadcast. One of the possible segmentations is based on the Fibonacci progression in order to limit the number of segments that the client has to receive simultaneously through periodic broadcast [1]. The segmentation is also fixed indepen-

dent of video popularity.

c) Multiple Proxies - Multiple Servers: We now extend this framework by including multiple geographically distributed servers. The way proxy servers operate and choose video prefixes is unchanged. Their role is to minimize the total amount of video data transferred over WAN. The requests for the part of the video not cached at the proxies have to be directed to one of the multiple available servers. The set of videos is partition among the servers in such a way that each video has to be stored in at least one server. On the other hand, there can be multiple replicas of a single video present in the system. Hence, there are two problems to solve. One is the video location problem, which includes assigning each video to one or more servers. The other is partitioning the requests for a given video among the servers that have a replica of this video. The goal is to minimize the WAN bandwidth as well as the aggregate I/O bandwidth requirements at servers. In order to achieve this goal we examine the dependency of I/O bandwidth and WAN bandwidth on the number of servers and their locations in the next section.

TABLE I
NOTATION

M	number of communities
N	number of videos
P	number of server locations
α_i^j	popularity of video j in community i
b_i^j	proxy $i's$ prefix size for video j
a_l^j	server $l's$ prefix size for video j
C_i^j	relative size of part of video j not cached by proxy i
U_{il}^j	relative size of video j delivered by broadcast from server l to community i
m^j	length of video j
S_i^j	length of segment i of video j
$\|x\|$	number of segments in x
ξ_l	indicator whether there is a server at location l
δ_l^j	indicator whether video j is stored at location l
B_l	storage space available at server l
W_l	I/O bandwidth available at server l
y_{il}^j	indicator whether requests for video j from community i are directed to server l

B. Object Location and Delivery Problem

The problem of object location has been considered for a distributed Web server system [20]. Some of the solutions proposed are based on the graph theoretic approaches, more specifically *facility location problem* [21]. The problem is defined as follows. There is a set of locations at which facilities may be built at a given cost. Each client must be assigned to one of the facilities. The objective is to find the number of facilities and their locations in such a way that the total cost, including facility opening cost and demand-related cost, is minimized. The latter type of cost is defined by the demand incurred by a given client multiplied by the distance between the client and the facility. Often, the available facility locations are the the same as clients locations. There are two broad classes of facility location objectives. One objective is to minimize the average or total demand weighted distance between clients and facilities. The other is to minimize the maximum distance between any client and the facility to which the client is assigned. In some problem formulations, the objective is to locate a fixed number of facilities. The k-center and k-median problems are examples of such an approach in maximum and average class of objectives, respectively.

The facility location problems are generally NP-hard and, therefore, we have to resort to heuristic approaches. In [22] the object location problem is formulated as choosing locations for server replicas, where each server replica contains the whole set of objects. The requests from a client are directed to the closest server replica. In [23] the problem is stated slightly differently. A single object does not have to be replicated in all selected server locations. All objects are available in their origin servers and in addition may be available at other servers. The objective is to choose the placement for each object. Client's request for an object is always directed to the closest server that contains that object. In both problems the objective is to minimize some measures of the amount of network bandwidth required to satisfy the requests. The heuristic-based solutions proposed for these problems come in a few flavors. They include greedy algorithms that choose the location whose inclusion in the set of selected facilities yields the smallest cost, random selection algorithms, and popularity-based algorithms that choose facility near the client generating the greatest demand.

The settings for the distributed web server and distributed video server are similar, however, the characteristics of the object delivered in both systems are quite different. Hence, we can use the experience with the distributed web server only to some extend. The duration of the video delivery to the client is significantly longer than the delivery of a web object due to the video size and rate requirements although video can be considered as a web object. Video stream consumes significantly more bandwidth than Web traffic due to the size of the object. Video objects require significantly more storage space at the server. One of the key factors in the large scale video delivery system, beside network bandwidth requirements, are the I/O bandwidth requirements at the servers and at the clients. This factor is not considered in the web server system. Hence, the objectives for the distributed video server include not only minimization and load balancing of the network bandwidth consumption, but also *minimization and load balancing with respect to the I/O bandwidth*. We will show later that the influence the replication and load balancing have on the network bandwidth requirements is quite different from the influence on the I/O bandwidth requirements.

III. Server Number and Location Influence on Resource Requirements

We now examine how resource requirements depend on the number and location of the servers. We investigate what number of servers yields the optimal I/O bandwidth and network bandwidth requirements and how changing the locations of the servers (or selection of a subset of the server locations) affects

these requirements. The notation used throughout the paper is summarized in Table I.

A. I/O Bandwidth in A Distributed Server System

We first concentrate on the I/O bandwidth consumption and its dependence on the number of servers. We show that the total I/O bandwidth consumption is a non-decreasing function of the number of servers employed. In other words, concentrating requests for a given video at a smaller number of servers either decreases the total I/O bandwidth consumption or does not affect it. Which one is the case depends on the video popularity.

Consider a video with low local popularity accessed from a number of communities. If the requests are spread among a large number of servers, the global popularity of the video at each server is going to be also low. Thus, each server chooses a prefix equal to the video length and delivers the video through a unicast transmission. The total I/O bandwidth consumption is equal to the sum of I/O bandwidth consumed by unicast transmission at each server. If the requests are directed to a smaller number of servers, the resulting global popularity of the video may be high enough to warrant a smaller prefix selection. Thus some part of the video would be delivered through broadcast and the total I/O bandwidth consumption would be lower than the sum of all unicast-related I/O bandwidth requirements. It may also be the case that global popularity at a single server is still low. Then, the server prefix is still equal to the whole video length and the I/O bandwidth requirements are the same for a smaller as well as a larger number of servers.

We now show the dependence of the I/O bandwidth consumption on the number of servers in a more formal way. Let the video set at a server consists of N videos and assume that the requests come from clients in M different communities. The I/O bandwidth requirement at the server is defined as follows:

$$W_l = \sum_{j=1}^{N}(|m^j - a_l^j| + \sum_{i=1}^{M}\alpha_i^j U_{il}^j) \quad (1)$$

where: m^j is the length of video j, a^j is length of server prefix for this video and $|m^j - a_l^j|$ is the number of segments in server suffix. $|m^j - a_l^j|$ is equal to the number of broadcast channels and constitutes I/O bandwidth consumption due to PB transmission. α_i^j is a number of concurrent accesses to video j from community i. U_{il}^j is the relative size of a part of video j delivered through a unicast transmission from a server to community i and is formally defined as:

$$U_{il}^j = \begin{cases} 0 & \text{if } b_i^j \geq a_l^j \\ \frac{a_l^j - b_i^j}{m^j} & \text{otherwise} \end{cases} \quad (2)$$

where b_i^j is the size of prefix selected for video j by a proxy in community i and l is the server number. Then $\alpha_i^j U_i^j$ is interpreted as I/O bandwidth requirement due to unicast transmission by a server. For simplicity and without loss of generality, we assume that the transmission rate of each of the segments and the playback rate of each video is equal to 1. Server prefix is selected in such a way that for a given set of proxy prefixes, the I/O bandwidth requirement at the server (Equation 1) is minimized.

Consider two servers receiving requests for video j from different communities. Since we concentrate for now on a single video we drop the superscript j to simplify the notation. Let a_1 and a_2 denote server prefixes for this video. The total I/O bandwidth consumption for both servers is expressed as:

$$W_1 + W_2 = |m - a_1| + \sum_{i \in A_1}\alpha_i U_{i1} + |m - a_2| + \sum_{i \in A_2}\alpha_i U_{i2}$$

where: A_1 and A_2 are the sets of communities sending requests to server 1 and server 2, respectively, and such that $A_1 \cap A_2 = \emptyset$. We show that there exists a server prefix such that the I/O bandwidth requirement after consolidation of requests at one server is no larger than the aggregate requirement for two servers. Assume that the server prefix after consolidation is equal to the smaller of two server prefixes: $a = min(a_1, a_2)$. Then, the I/O bandwidth consumption due to this video after consolidation is equal to:

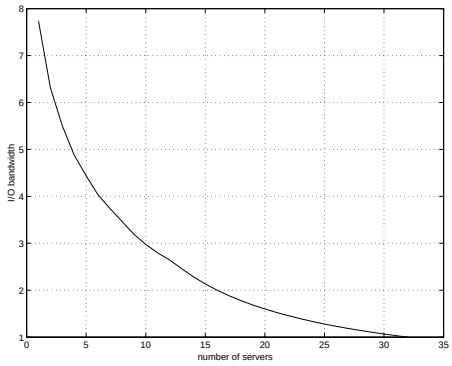
$$W_{12} = |m - a| + \sum_{i \in A_1 \cup A_2}\alpha_i U_{i(12)}$$

Notice that the I/O bandwidth consumption due to broadcast transmission of the video after consolidation is reduced by $|m - max(a_1, a_2)|$ or unchanged if $max(a_1, a_2) = m$. Assume now that $a_1 \leq a_2$. Then the total I/O bandwidth consumption due to unicast transmission of this video at both servers is equal to :

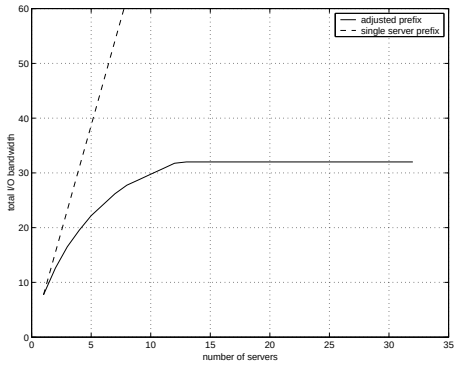
$$\begin{aligned} W_1^u + W_2^u &= \sum_{\substack{i \in A_1 \\ b_i < a_1}} \alpha_i \frac{a_1 - b_i}{m} + \sum_{\substack{i \in A_2 \\ b_i < a_2}} \alpha_i \frac{a_2 - b_i}{m} \\ &= \sum_{\substack{i \in A_1 \\ b_1 < a_1}} \alpha_i \frac{a_1 - b_i}{m} + \sum_{\substack{i \in A_2 \\ b_i < a_1}} \alpha_i \frac{(a_1 - b_i + a_2 - a_1)}{m} \\ &\quad + \sum_{\substack{i \in A_2 \\ a_1 \leq b_i < a_2}} \frac{(a_2 - b_i)}{m} \\ &\geq \sum_{\substack{i \in A_1 \cup A_2 \\ b_i < a_1}} \alpha_i \frac{a_1 - b_i}{m} = W_{12}^u \end{aligned}$$

We observe that, the unicast I/O bandwidth consumption is reduced, or unaffected if $a_1 = a_2$. Since neither unicast nor broadcast I/O bandwidth requirements increase, we conclude that the total requirements do not increase when the number of servers decreases. Moreover, the I/O bandwidth usage decreases if $max(a_1, a_2) < m$ and/or $a_1 \neq a_2$. This result is fairly intuitive. For communities whose proxy prefix is larger than a_1, but smaller than a_2, the unicast transmission from server 2 is now replaced with the broadcast transmission from server 1.

The following example illustrates the aforementioned behavior. Consider a number of communities having the same access profile for a given video and with homogeneous proxies. The requests for the video are distributed equally among a number of servers. As we vary the number of servers from 1 to the number of communities, we observe that the global popularity of the



(a) I/O bandwidth required for a single server as a function of the number of servers



(b) Total I/O bandwidth as a function of the number of servers

Fig. 3. Server number influence on the I/O requirements

video at each server decreases and consequently the server prefix size increases. Figure 3(a) shows that the I/O bandwidth consumption at each server decreases since the load is divided among an increasing number of servers. Figure 3(b) shows that the total I/O bandwidth consumption *increases*. The Figure illustrates two cases: when the server prefix size is fixed, and when each server chooses an optimal prefix size. In the first case the I/O bandwidth requirements increase much faster. An optimal prefix size choice allows to slow down that increase considerably. Notice that when the number of servers reaches 13, the total I/O bandwidth remains unchanged with the further increase in the number of servers. At this point server prefix is equal to the video length and further change is not possible.

In summary, the I/O bandwidth requirements are always optimal in a single server system, independent of that server's location. Increasing the number of servers may result in an increase of I/O bandwidth requirements provided that the global popularity of a video is high enough. For low global popularity video there is no difference between a single and multiple server system in terms of I/O bandwidth. If the number of servers does make a difference, the server selection for given communities may also matter. It does matter if the local video popularity varies a lot from one community to another. In that case, it is beneficial to direct the requests from the communities, where the video is very popular, to a single server. In other words, it is beneficial to "build" high global popularity at one server.

B. WAN Bandwidth in Distributed Server System

We examine how network bandwidth requirements depend on the number and locations of servers. WAN bandwidth for a single server is defined in the following way:

$$R_l = \sum_{j=1}^{N} \sum_{i=1}^{M} \alpha_i^j C_i^j d_i \tag{3}$$

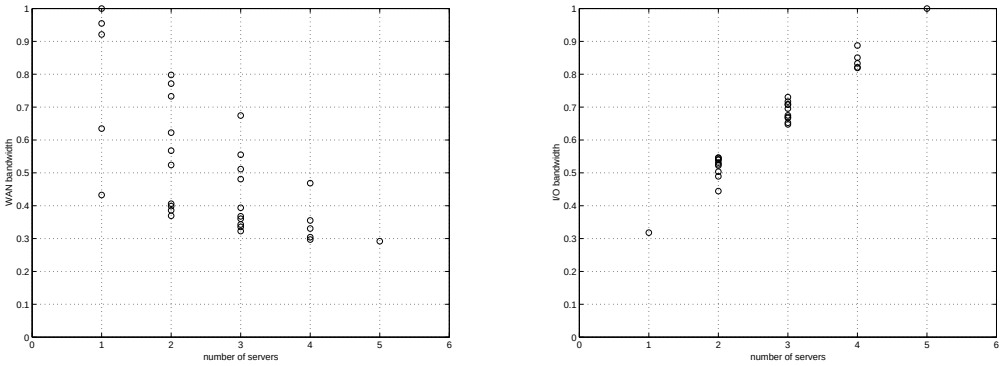
where d_i is the distance from community i to the server. C_i^j is the relative size of the part of video j not cached by a proxy in community i. It is determined by the size of the proxy prefix and is formally defined as:

$$C_i^j = \frac{m^j - b_i^j}{m^j} \tag{4}$$

where b_i^j is the proxy prefix selected for video j by proxy i in such a way that the amount of data delivered to clients by a server ($\sum_{j=1}^{N} \sum_{i=1}^{M} \alpha_i^j C_i^j$) is minimized. WAN bandwidth is proportional to the amount of data that has to be transmitted by a server to each community and to the distance between the community and the server. Note that we are assuming both segment (channel) transmission and video playback rates are normalized to 1. Due to its dependence on the distance, WAN bandwidth requirement is a decreasing function of the number of servers. It exhibits such behavior if we assume each community is assigned to the closest server available. Otherwise, it is possible that a smaller number of servers results in smaller network bandwidth requirements. The total WAN bandwidth consumed depends very strongly not only on the number of servers, but also on their locations. A decreasing character of the network bandwidth function is quite intuitive. Assume that first a single server is used to deliver all videos to all communities. Then another server is added. The requests from some of the communities are redirected to the second server but only if their distance to that server is smaller than the distance to the first one. The total amount of data transferred over WAN is the same independent of the number of servers, but the distance decreases (or at least does not increase) for a higher number of servers.

Figure 4(a) presents an example of network bandwidth consumption as a function of the number of servers for a single video. With 5 potential locations, there is a number of ways to choose a smaller than 5 number of servers (C_n^i). For examples, there are 10 different ways to choose 2 out of 5 servers. Hence, for a given number of servers on x-axis there is one point plotted for each possible combination. The network bandwidth value is shown as a percentage of the maximum value in each case to illustrate the variation of network bandwidth requirements.

We observe that network bandwidth generally decreases with the number of servers. The lowest network bandwidth consumption is observed in a 5-server system, the highest in a



(a) WAN bandwidth

(b) I/O bandwidth

Fig. 4. Server number and selection influence on the resource requirements

1-server system. For a given number of servers, the network bandwidth value obtained varies quite significantly depending on the locations of the selected servers. This effect is especially visible in single server systems and becomes less profound as the number of servers increases. Figure 4(b) presents corresponding I/O bandwidth requirement. In contrast to network bandwidth, I/O bandwidth requirements increase with the number of servers and show much weaker dependence on the server locations.

IV. Problem Formulation

We use the term server to refer not to a machine but rather to a location, where a server cluster can be deployed. Given the geographical distribution of resources, the task is to *distribute the video set and the client requests among available servers*. We first describe briefly video delivery framework under consideration and then proceed to formally formulate the problems.

We consider a set of Constant-Bit-Rate videos available from a set of geographically distributed servers. The possible server locations are placed strategically among communities but their number is much smaller than the number of communities. The distance between each community and each server is given by some metric expressing network delay or the number of routers along the path for example. The goal is to choose locations for a fixed number of servers out of all available locations, distribute the video set among the selected servers and distribute requests for each video among all serves holding this video in such a way that resource requirements are minimized. Each video has to be stored on at least one server but there may be multiple replicas of a given video available in the system. That is, a video may be stored at multiple servers. Since all clients within one community have the same distance to a given server and for simplicity we assume that all requests for a given video from a single community are directed to a single server. It is possible, on the other hand, that different videos are delivered to a given community by different servers.

We approach the problem by assigning the clients to the servers on a per-video basis at the community level, i.e., we choose a *server for each (video, community) pair*. Notice that such assignment determines not only the request distribution, but also the video location, i.e., the number of replicas for each video and their locations. If a server is not selected to accept requests for a given video from any community, it does not keep a copy of that video. Otherwise, a replica of the video is stored at that server.

We consider two problem formulations. The first formulation is an uncapacitated problem, in which we assume that each server has adequate resources to accept all requests and the goal is to use these resources efficiently. The other is a capacitated problem, in which the resources available at each server are limited and hence, the objective is to maximize the number of accepted requests. The resources include I/O bandwidth, network bandwidth and storage space.

A. Uncapacitated Problem

We proceed now to formally define the uncapacitated problem. Let P be the number of possible locations for geographically distributed servers. The distance between each server location and each community is captured by the distance matrix d_{il}, $(1 \leq i \leq M,\ 1 \leq l \leq P)$. We introduce a set of variables y_{il}^j to denote the server selection for each video and each community. y_{il}^j is equal to 1 if community i requests video j from server l and is equal to 0 otherwise. Our objective in the uncapacitated problem is to minimize the total I/O bandwidth and WAN bandwidth consumption. More formally:

$$\begin{array}{ll} \text{minimize} & K_1 \sum_{l=1}^{P} \sum_{j=1}^{N} \sum_{i=1}^{M} \alpha_i^j C_i^j d_{il} y_{il}^j + \\ & K_2 \sum_{l=1}^{P} \sum_{j=1}^{N} (|m^j - a_l^j| + \sum_{i=1}^{M} \alpha_i^j U_{il}^j y_{il}^j) \\ \text{s.t.} & 1) \sum_{l=1}^{P} \xi_l = k \\ & 2) \sum_{l=1}^{P} y_{il}^j = 1 \ \ i = 1, ..., M \text{ and } j = 1, ..., N \\ & 3) \ y_{il}^j \in \{0, 1\} \end{array} \tag{5}$$

The first component of the optimization function, the network bandwidth consumption, depends on two factors: the amount of data transmitted from the server (not cached at the proxy) and the distance at which it is transferred. The second component accounts for the total I/O bandwidth consumption at servers. a_l^j denotes the video prefix selected for video j by

server l. For details on proxy prefix b_i^j and server prefix a_l^j selection see [4]. The magnitude of the WAN bandwidth consumption may be very different from that of I/O bandwidth consumption. Hence, two parameters, K_1 and K_2 are used to scale their values. By adjusting the values of K_1 and K_2 more weight can be put on either network or I/O bandwidth requirements.

The constraints ensure that: 1) the total number of servers selected is k and the total number of server locations is P, 2) for a given community and a given video only one server is selected, and 3) all requests for a video from a community are directed to a selected server. ξ_l in the first constraint indicates whether location l was selected for a server or not, that is, if the requests from at least one community for any video are directed to that location, and is formally defined as:

$$\xi_l = \begin{cases} 1 & \text{if } \sum_{i=1}^{M}\sum_{j=1}^{N} y_{il}^j > 0 \\ 0 & otherwise \end{cases} \quad (6)$$

The second constraint ensures also that every video which is requested by at least one community is available in at least one location. This condition is covered by a stronger requirement that all requests are directed to some server. Since the request distribution determines the videos distribution, the video availability condition is satisfied automatically.

B. Capacitated Problem

The emphasis of the capacitated problem formulation is on the request distribution among a number of pre-selected servers with different and limited capacities. We assume that the number of servers, their locations and capacities are given. The request distribution is defined by a set of aforementioned variables $\{y_{il}^j\}$. Ideally all requests from all communities can be accepted. Given limited resource availability, the goal is to maximize the number of requests that can be accepted. Notice that the I/O bandwidth requirement function expresses the average consumption, since it is based on the average number of concurrent accesses to a video. Thus, we minimize the probability that a request is rejected by *maximizing the minimum spare I/O bandwidth capacity* at the servers. Storage space requirement on the other hand, is fixed for a given request distribution. Therefore, it is sufficient to require that the storage capacity is not exceeded. The second goal is the same as in the uncapacitated problem: minimizing the total network bandwidth consumption. The capacitated problem is defined more formally in the following way:

$$\begin{array}{ll} \text{min.} & K_1 \sum_{l=1}^{k}\sum_{j=1}^{N}\sum_{i=1}^{M} \alpha_i^j C_i^j d_{il} y_{il}^j + \\ & K_2 \max_{l=1}^{k}(\sum_{j=1}^{N}(|m^j - a_l^j| + \sum_{i=1}^{M} \alpha_i^j y_{il}^j U_{il}^j) - W_l) \\ \text{s.t.} & 1) \sum_{j=1}^{N} \delta_l^j m^j \le B_l \\ & 2) \sum_{l=1}^{k} y_{il}^j = 1 \;\; i = 1, ..., M \\ & 3) \; y_{il}^j \in \{0, 1\} \end{array} \quad (7)$$

where B_l denotes storage space and W_l denotes I/O bandwidth available at server l. δ_l^j indicates whether video j should be stored at server l or not. It is a function of request distribution defined as follows:

$$\delta_l^j = \begin{cases} 1 & \text{if } \sum_{i=1}^{M} y_{il}^j > 0 \\ 0 & otherwise \end{cases} \quad (8)$$

Constraints 2) and 3) are similar to the constraints of the uncapacitated problem. They ensure that *all* requests are directed to some server and that the assignment is done on per (video, community) basis. In contrast to uncapacitated problem, this time we consider a fixed number of servers k and require that the storage capacity is not exceeded at any of the servers. Therefore, the total storage space available must be equal to at least the sum of the sizes of all videos for a solution to exist: $\sum_{l=1}^{k} B_l \ge \sum_{j=1}^{N} m^j$.

C. Capacity Distribution

The capacitated problem formulation of optimizing the total resource consumption leads to the second group of problems in the geographically distributed video server system, that is, the *resource availability at each server location*. We explore the problem of assigning I/O capacity and storage space to each location. The network bandwidth is not included in the consideration since its consumption it not localized only at servers but distributed in the network along paths between clients and servers.

The intuitive approach is to place the capacity where the requirements are in order to save the WAN bandwidth and to improve the performance. Therefore, we first identify which components of the system, namely which communities generate the most demand. The first set of candidates are the communities which are large in terms of number of clients. The second set of candidates are diverse communities, in which the video demand distribution is less skewed. Recall that due to the nature of I/O bandwidth requirement function, given a fixed number of requests, the requirements are lower if the majority of requests are for a small number of videos. Thus, there are two parameters that characterize a community with respect to the demand it generates: *size and the degree of diversity*.

Our goal is to choose the *number and locations* of the servers, and assign the I/O capacity and the storage space to each selected server. We assume that the total amount of both types of resources is limited and that there is a fixed opening cost for each server location. The demand-based cost of a given resource distribution ($\{W_l, B_l\}$) is given by the network bandwidth required to satisfy the demand. We formulate the capacity assignment problem as the minimization of the total cost, including opening cost and demand-based cost, subject to resource availability.

We relate the network bandwidth required for a given capacity distribution to the community characteristics by the use of a hypothetical video set. Given a number of videos, the popularity of each video in a given community is drawn from a Zipf distribution [24,25], whose mean is determined by the community size and the skewness parameter is determined by the degree of diversity: $\alpha_i^j = s_i (r_i^j)^{t_i - 1}$, where t_i denotes the skewness parameter for community i and s_i denotes its size. Given the video popularity for each community, we assign each (video, community) pair to one server location in such a way that if the resource requirements determined by this assignment were used to distribute the capacity, the total cost would be minimized:

$$\begin{array}{ll} \text{min.} & \sum_{l=1}^{P}\left(\sum_{j=1}^{N}\sum_{i=1}^{M}\alpha_i^j C_i^j d_{il} y_{il}^j + \xi_l O\right) \\ \text{s.t.} & 1)\ \sum_{l=1}^{P}\sum_{j=1}^{N}(|m^j - a_l^j|) + \sum_{i=1}^{M}\alpha_i^j y_{il}^j U_{il}^j) \le W \\ & 2)\ \sum_{l=1}^{P}\sum_{j=1}^{N}\delta_l^j m^j \le B \\ & 3)\ \sum_{l=1}^{P} y_{il}^j = 1 \ \ i = 1, ..., M \\ & 4)\ y_{il}^j \in \{0,1\} \end{array} \tag{9}$$

where O is the opening cost. Requests distribution is represented by the set of variables y_{il}^j as defined previously. The components of the optimization function include the total network bandwidth required and the opening cost. Constraints 1) and 2) ensure that the total resource requirements, I/O bandwidth and storage space, respectively, do not exceed the amount available for distribution. The remaining two constraints ensure, similarly to two previous problems, that all requests are directed to some server. Given solution to this optimization problem $\{y_{il}^j\}$, we assign the capacity in the following way:

$$W_l = \sum_{j=1}^{N}(|m^j - a_l^j| + \sum_{i=1}^{M}\alpha_i^j y_{il}^j U_{il}^j), \quad B_l = \sum_{j=1}^{N}\delta_l^j m^j$$

V. Heuristic Solutions

A. Heuristics for Uncapacitated Problem

The constrained problem of selecting a fixed number of servers from all available locations is NP-complete [26]. Thus, we devise heuristic solutions to the uncapacitated problem. Our solution consists of three steps. Proxy prefix for each proxy and each video b_i^j is selected in the first step in such a way that the total amount of data that has to be transferred from a server to a client is minimized, subject to the buffer size and I/O bandwidth constraints at the proxy. Then, given proxy prefixes we choose a server for each video and each community in such a way that the total I/O and network bandwidth requirements of all servers are optimized. Finally, given server assignment for each video and each community and proxy prefixes, we select the server prefix for each server and each video in its set in such a way that the minimization of I/O bandwidth consumption at each server is achieved. In the following description we concentrate on the second step, i.e., video location problem assuming that the proxy prefixes for each video are already selected (for details on proxy and server prefix selection see [4]).

In order to direct requests for each video from each community to k servers we propose a *greedy approach*. There are two criteria determining how good a solution is: I/O bandwidth and WAN bandwidth consumption. Recall that WAN bandwidth consumption depends on the amount of data transferred between a client and a server and the distance over which it is transferred. The amount of data is determined already at this point by the proxy prefix size. Hence, in order to minimize the WAN bandwidth consumption the logical choice for each community is to direct its requests to the closest server location. The higher the number of servers available, the smaller the total WAN bandwidth requirements. On the other hand, the optimal I/O bandwidth consumption is achieved with only one server independent of its location. I/O bandwidth is not a good criterion for a greedy selection of k servers since there is no incentive to choose more than one server. Thus, we perform greedy server selection based on the WAN bandwidth consumption first, and then adjust the solution by redirecting some of requests to control the I/O bandwidth consumption.

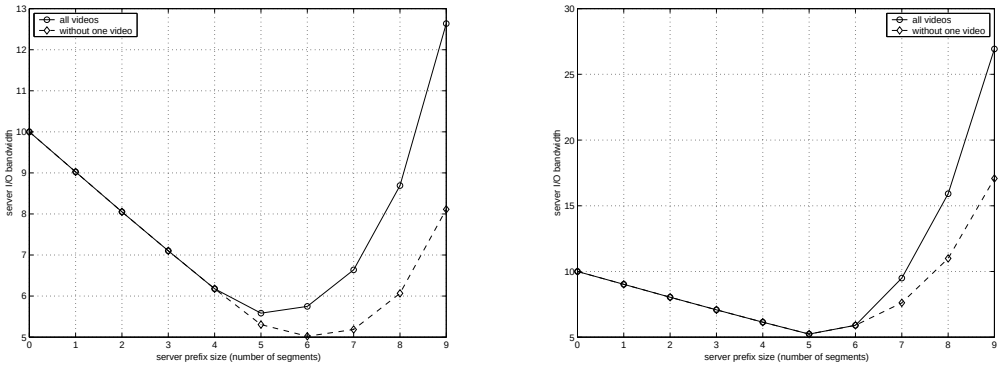
1) Greedy Server Selection: The greedy algorithm constructs a solution by choosing one server at a time making a locally optimal decision. The first server denoted by s_1 is selected in such a way that if all requests were directed to a single server, the network bandwidth requirement would be optimal for s_1. Next, another server s_2 is added to the set of selected servers in such a way that the network bandwidth consumption for a pair (s_1, s_2) is equal to the minimum over all pairs (s_1, l), $l = 1, .., P$ and $l \neq s_1$. In a similar way we choose s_3 and the consecutive servers until the number of servers is equal to k. At each step the requests from each community are directed to the closest server in the selected server group.

2) I/O Bandwidth Adjustment Levels: The greedy algorithm yields a solution whose I/O bandwidth requirements may be quite far from optimal. Depending on the relative importance of I/O bandwidth requirements, the solution can be adjusted to lower these requirements at the cost of potentially increase of the total WAN bandwidth consumption. The adjustment can be made at two levels of granularity. We can redirect *all requests* for a given video from one server to another, which is equivalent to removing the video from one server's video set. Or we can redirect requests for a video *from some communities only* from one server to another. We refer to the server from which requests are redirected as a *source* server, while a server to which requests are redirected as *destination* server.

a) Coarse level of adjustment: We examine the more coarse level of granularity first. The video is removed from source server's video set. Notice that if the video is *not* in the destination server's video set already, the change will not affect the total I/O bandwidth consumption and only increase the network bandwidth consumption. Thus, the goal is to consolidate requests reaching two different servers. The I/O bandwidth consumption at the source decreases after consolidation while the I/O bandwidth consumption at the destination may increase. Recall that consolidating requests at one server cannot increase the total I/O bandwidth consumption.

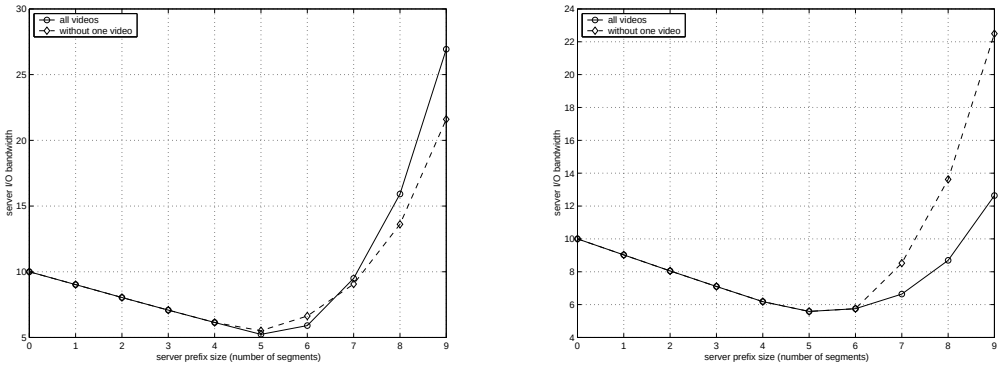
b) Fine level of adjustment: At the second level of granularity the I/O bandwidth requirement adjustment is performed by redirecting requests originating in one community for a given video from one server to another. In this case the video may remain in the source's video set. In contrast to the more coarse level of adjustment granularity, such change may yield an increase of the total I/O bandwidth consumption. Thus, we formulate now the necessary and sufficient conditions for such a redirection to reduce the total I/O bandwidth consumption.

Consider requests for video j originating in community i. The effect of redirecting these requests *from* server l on the I/O bandwidth consumption at this server depends on the relation between the video proxy prefix size b_i^j for community i and the server prefix size a_l^j. If proxy prefix is larger than the server prefix, redirecting requests from the server does not affect the I/O bandwidth consumption at the source. Part of the video received by clients from the server is transmitted in broadcast



(a) proxy prefix $<$ server prefix (b) proxy prefix $>$ server prefix

Fig. 5. Server I/O bandwidth consumption after requests redirection *from* a server



(a) proxy prefix $<$ server prefix (b) proxy prefix $>$ server prefix

Fig. 6. Server I/O bandwidth consumption after requests redirection *to* a server

mode whose I/O bandwidth consumption does not depend on the number of clients. Also the server prefix size does not change meaning that the optimal I/O bandwidth consumption at the server is still obtained with the same prefix size. The proof of such behavior is presented in the Appendix. Hence the necessary condition for the redirection to *decrease* the I/O bandwidth requirements at the source is that *proxy prefix is no larger than the server prefix*. It follows naturally that the server prefix must be larger than 0. Figure 5 shows the destination server I/O bandwidth consumption as a function of server prefix size before and after the redirection of requests from one community for a video with proxy prefix smaller than server prefix 5(a), and for a video with proxy prefix larger than server prefix 5(b). We observe that in the first case the minimum I/O bandwidth is lower and obtained for a larger server prefix size, while in the latter case the optimal value does not change and the server prefix size remains the same.

Similarly when the requests for a video originating in a given community are redirected *to* server l from another server, the I/O bandwidth consumption at the destination increases only if the proxy prefix is smaller than the server prefix. In summary, the total I/O bandwidth consumption may be reduced only if we redirect requests for a video from a server whose prefix is larger than the proxy prefix. If in addition the destination server has prefix smaller than the proxy prefix, the total I/O bandwidth consumption is reduced. Otherwise, it is possible that the reduction at the source is smaller than the increase at the destination server. The first condition is a necessary condition for the reduction of the *total* I/O bandwidth consumption while the combination of both conditions is a sufficient condition for such reduction. Notice that if a source server receives requests for a given video only from one community, the necessary condition is satisfied. In this case the proxy prefix is no larger than the source server prefix and the I/O bandwidth consumption at the source is always reduced.

3) I/O Bandwidth Adjustment Heuristic: Based on these observations, we can lower the total I/O bandwidth requirements by redirecting some requests form one server to another. The heuristic algorithm consists of two operations: selecting the source and destination servers and selecting the object, video or (video, community) pair, to be redirected. In order to minimize the number of redirections, we can choose source, destination and object that yield the largest reduction of the I/O bandwidth. However, the complexity of identifying such candidates is quite high for a large number of servers and large number of videos $\mathcal{O}(k^2M)$. Thus, we formulate a greedy algorithm. We consider

all videos that are available on at least two servers. From this set we choose the video whose global popularity is the highest and consolidate requests for this video from two servers with the largest I/O bandwidth consumption due to this video. We choose the server for which the increase in network bandwidth consumption is the smallest as the destination. Such operation is repeated until a desired I/O bandwidth requirement value is reached. Notice that after a sufficiently large number of steps the minimal value can be reached when each video is available at one server only. A more precise adjustment is performed at a finer level of granularity by choosing a (video, community) pair, which satisfies the necessary condition for the reduction of the total I/O bandwidth at the source and has the largest I/O bandwidth requirement.

4) Load Balance Heuristic: The I/O bandwidth requirements can be modified also to balance the loads across all servers. For that purpose we use the solution yielded by the greedy server selection as the starting point. Server with the maximum I/O bandwidth consumption is selected as a source and the server with the smallest I/O bandwidth consumption as a destination. We redirect requests for the video with the highest I/O bandwidth requirements at the source such that the redirection does not increase the destination's I/O bandwidth consumption beyond the average value for all the servers. This step is repeated until the difference between the I/O bandwidth requirements and the average value at each server does not exceed some predefined value.

B. Heuristics for Capacitated Problem

A similar approach can be used to obtain a solution for the capacitated problem. We assume now that each server has a limited I/O bandwidth capacity and our goal is to maximize the total number of requests that can be satisfied. Initially we direct requests from each community to the closest server in order to obtain a solution which is optimal with respect to the network bandwidth requirements. Next, we modify the I/O bandwidth requirements to maximize the minimum spare capacity at any server. Our heuristic achieves that goal by equalizing the spare capacity across all servers. We redirect requests from a server with the highest difference between the I/O bandwidth requirements and its capacity to a server with the highest spare capacity. The video for which the requests are redirected is selected as a video with the largest I/O bandwidth requirements at the source and the redirection does not increase the destination's I/O requirements beyond its capacity. If none of the videos at the source satisfy the latter condition, we choose the second largest overloaded server as the source. These steps are repeated until no more redirections are possible. In order to maximize the number of requests that can be satisfied, we next perform the adjustment at a finer granularity. We choose a (video, community) pair with the maximum I/O bandwidth requirement at the source and the redirection does not increase destination's I/O bandwidth requirements beyond its capacity.

C. Capacity Distribution Heuristic

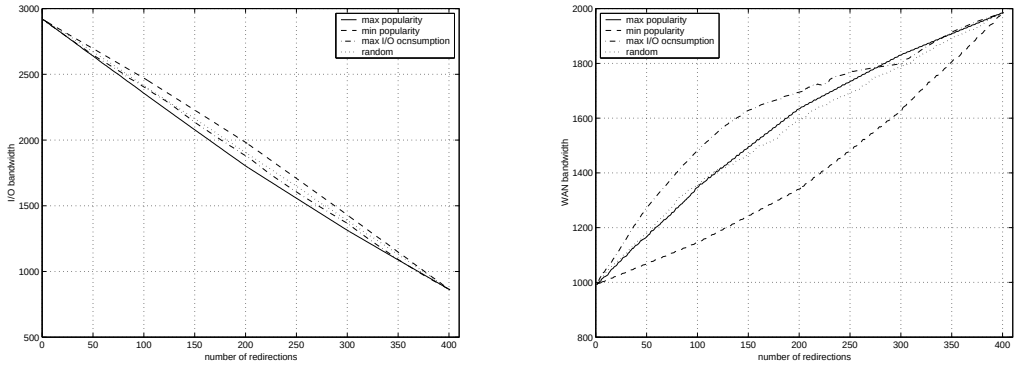
Similarly to the uncapacitated request distribution problem, capacity distribution is also an NP-hard problem. Therefore, we devise a heuristic solution also for this problem. The total opening cost increases with the number of servers while the demand-based cost given by the network bandwidth consumption decreases. Given this trade-off, we are looking for a number of server locations that achieve the minimum total cost. The I/O bandwidth and storage space constraints limit the number of servers in which a video can be stored.

We solve the problem in two steps. First we choose the servers in a greedy way to obtain a sub-optimal value of the cost function ignoring resource availability constraints. In this phase the requests are directed always to the closest server containing a video in order to minimize the demand-based cost. Next we adjust the solution to account for the resource constraints.

1) Greedy Server Selection: The minimum network bandwidth consumption is achieved with all P servers utilized, by directing all requests from each community to the closest server. This set of servers results also in the highest opening cost. As the number of servers decreases, the opening cost decreases while the network bandwidth requirements increases (more specifically do not decrease). We search for the server set yielding the optimal cost in a greedy way. From the set of all servers we eliminate a server, whose removal yields the smallest increase in the network bandwidth. Since the opening cost depends only on the number of servers, the server choice does not affect the corresponding decrease in the total opening cost. We repeat this process until there is only one server left in the set.

2) I/O and Storage Capacity Distribution: The server set is selected in the first step under the assumption that each community sends all of its requests to the nearest server. Such request distribution determines I/O bandwidth and storage space requirements. These requirements may violate the resource availability constraints. Therefore, we adjust the solution by redirecting some requests as necessary. The goal for this phase is to satisfy the I/O constraints with the minimal increase in cost. Redirection may only *decrease* the total opening cost. However, any redirection results in an increase in the total demand-based cost. Therefore, we use a strategy similar to the heuristic I/O bandwidth adjustment in uncapacitated problem. We choose a video with the minimum global popularity and two servers with the maximum I/O bandwidth consumption due to the selected video. The server for which the increase in network bandwidth consumption is smaller becomes the destination and the other server becomes the source. The process is continued until I/O bandwidth constraint is satisfied. As we will shortly see choosing videos with low popularity for request redirection result in lower increase in cost than using other criterion such as the highest popularity or the highest I/O bandwidth consumption.

Notice that each redirection reduces storage space requirements at the source server and does not increase this requirement at the destination. If the solution satisfies the total I/O bandwidth constraint, but violates the storage space constraint, we continue the redirection process until the storage constraint is also satisfied. Notice also that in contrast to the capacitated problem, we do not maximize the spare capacity at each location. The goal is to place the capacity close to the demand. Therefore, we make only adjustments that are necessary to satisfy the availability constraints. Any redirection beyond this



(a) I/O bandwidth distribution

(b) WAN bandwidth distribution

Fig. 7. Influence of video choice on the resource requirements

point would increase the cost unnecessarily.

VI. Numerical Test Results

We have conducted a number of tests to examine the properties of the proposed heuristic algorithms. In the evaluation we concentrate on the main resource requirements, i.e., I/O bandwidth and WAN bandwidth.

A. Test settings

There are 100 communities whose locations determining the distance from the server locations are selected at random. The set of video contains 100 equal-length videos. The popularity for each video is drawn from a Zipf distribution: $\alpha_i^j = s_i(r_i^j)^{t-1}$, where $t = 0.27$ [27] , s_i is the size of community i (i.e., the number of clients) drawn from a normal distribution and r_i^j is the popularity rank of video j in community i. Notice that since all videos are of the same size, the popularity drawn from a Zipf distribution expresses the number of concurrent accesses. The ranks for all 100 videos are changed for each community by a permutation. More precisely we divide the set of videos into 4 groups and apply permutation to each group separately. We assume that there are videos, which are generally popular, although their exact ranks may still vary from one community to another. We choose 10 locations at random as potential server locations.

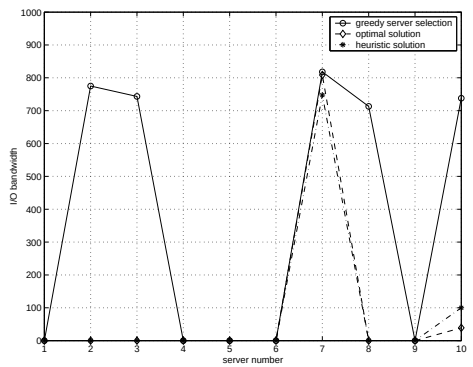


Fig. 8. I/O bandwidth distribution comparison

B. Uncapacitated Problem

The greedy server selection algorithm chooses 5 out of 10 server locations. The heuristic that we use to adjust the I/O bandwidth requirements can yield any value between the solution obtained with the greedy server selection and the optimal I/O bandwidth requirements. The heuristic can obtain the optimal value after performing sufficiently large number of redirections. A stop criterion can be defined to obtain a desired value of I/O bandwidth requirements larger than the minimum. Notice that in order to reach the optimal value only adjustments at the more coarse level of granularity have to be performed.

1) Video Selection Criterion: We have conducted a number of tests to examine how the selection of the objects, and the source and destination servers affect the I/O bandwidth requirements. We use the global popularity of a video and the I/O bandwidth consumption due to a video as the two possible criteria for the object selection. Figure 7(a) shows an example of how the total I/O bandwidth consumption changes with the number of redirections for the maximum and minimum version of both criteria. We have included also a random video selection for the purpose of comparison. In each case two servers with the maximum I/O bandwidth consumption due to the selected video are chosen as the source and destination servers. The fastest reduction is obtained when we select the video with the highest global popularity at each step, and the slowest occurs when the lowest popularity video is selected. Figure 7(b) presents the corresponding increase in the network bandwidth requirements. After examining a number of cases, we conclude that the fastest reduction of I/O bandwidth is obtained either with the highest popularity used as a criterion or the highest I/O bandwidth requirements. The difference between different criteria, however, is not large. We observe also that changing the criteria for the selection of the source and destination servers does not have a significant influence on the speed of I/O bandwidth requirements reduction.

2) Network Bandwidth Increase: Each redirection increases network bandwidth consumption. We control this increase by consolidating requests always at the server for which the network bandwidth consumption increase is smaller. The lowest network bandwidth requirement for the solution optimal from

the I/O bandwidth point of view, is obtained by placing each video at a server, which yields the smallest network bandwidth consumption. Note that the number of servers used by such solution may be smaller than the targeted number k. The heuristic makes a greedy choice of destination server for each redirection. Therefore, the solution with optimal I/O bandwidth requirements may not reach the lowest possible network bandwidth requirement. In order to evaluate the solution given by the heuristic w.r.t. network bandwidth, we compare this solution with the lowest network requirements obtainable without increasing the I/O bandwidth consumption. Figure 8 presents the comparison between resource requirements at each server for the greedy server selection, the optimal I/O solution with the lowest network bandwidth requirements and the optimal I/O solution obtained with the heuristic. We observe that servers 2, 3, 7, 8 and 10 are selected by the greedy approach. The I/O bandwidth adjustment heuristic reduces this set to only two servers 7 and 10. The same servers are chosen for the optimal solution with minimal WAN bandwidth consumption. Although these two solutions differ in distribution of requests among servers, their total resource requirements are very close to each other. We conclude that the heuristic solution allows us to adjust the I/O bandwidth to an arbitrary level limited only by the minimum I/O bandwidth requirements while controlling the increase in network bandwidth requirements.

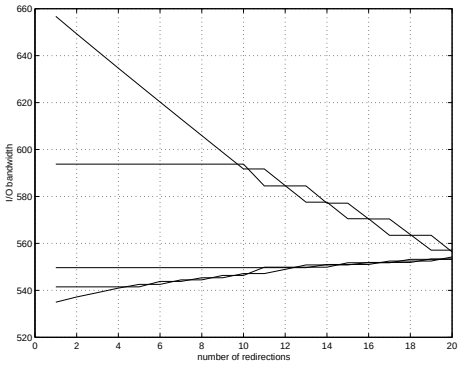


Fig. 9. Load balance heuristic results

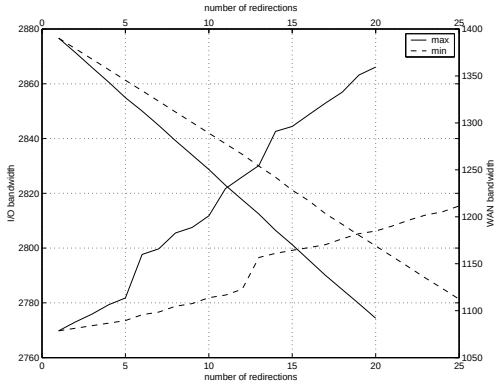


Fig. 10. Video selection for load balance heuristic

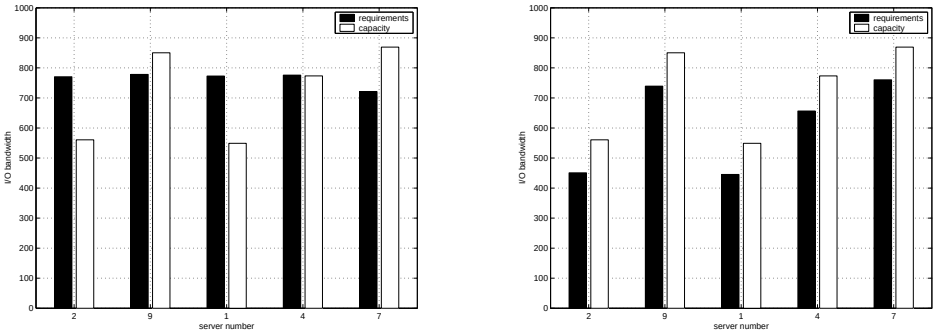
C. Load Balance

Figure 9 presents the results of load balancing heuristics. Each curve corresponds to the I/O bandwidth requirements at one server and shows how these requirements change with the number of redirections. We observe that the requirements at two out of the five servers are above the average. At each step the I/O bandwidth consumption at one of these two servers is reduced while I/O bandwidth consumption of one of the remaining servers is increased. The process continues until the difference between the maximum requirement and the minimum requirement is smaller than some predefined value. We observe also that the total I/O bandwidth requirements are reduced as a result of the redirections. However, not every step yields such a reduction. A redirection may be performed on the requests for the video that is not present on the destination server prior to the redirection. The number of steps required to achieve load balance depends on the variance of the I/O bandwidth of the solution given by the greedy server selection and to a lesser degree on the predefined value used as a stop criterion.

In order to minimize the number of redirections required to achieve load balance, the object selected in each step is the video with the largest I/O bandwidth requirement at the source server. The high I/O bandwidth requirements are related to high popularity. As a result of using such criterion we observe that the number of copies of the most popular video is reduced compared to less popular videos. We have conducted a test in which the video with the smallest I/O bandwidth requirements at the source server is selected for request redirection. As a result, the load balance was achieved after a larger number of steps. We observe also that since in the case, the number of copies of the least popular videos is reduced, the network bandwidth requirements are lower than for the solution obtained previously. Figure 10 presents the comparison of the total I/O and WAN bandwidth requirements as a function of the number of redirections in both cases. We observe that the total I/O bandwidth consumption achieved by load balance heuristic is lower when a video with the maximum I/O bandwidth requirements is selected in each step, but the difference is not very significant. The difference in network bandwidth consumption in these two cases, on the other hand, is quite large. Therefore, we conclude that there is a trade-off between the number of steps required to achieve load balance and network bandwidth requirements resulting from the heuristic solution.

D. Capacitated problem

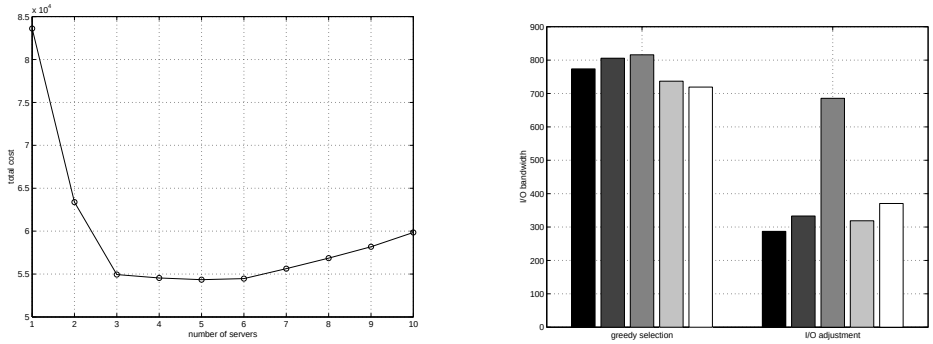
The first step in the heuristic for the capacitated problem is directing requests to the closest server. Figure 11(a) presents an example of the resulting capacity distribution and I/O bandwidth requirement distribution. We observe that not only the capacity of some of the servers is exceeded but also that the total requirements exceed the total capacity. By redirecting the requests we achieve two goals: bring the *total* and *per-server* I/O bandwidth requirements below the capacity if possible. Figure 11(b) shows the capacity and I/O bandwidth requirements distribution after redirection processes. Recall that our goal is not only not to exceed the capacity at any server but also to maximize the minimum spare capacity on each server. Therefore,



(a) Requirement distribution before adjustment

(b) Requirement distribution after adjustment

Fig. 11. Capacitated problem heuristic results



(a) Greedy server selection solution

(b) Capacitated problem solution

Fig. 12. Capacity distribution

the redirection process ends when the spare capacity at each server does not differ from the average spare capacity by more than a predefined value.

E. Capacity distribution

Figure 12(a) illustrates the total cost for the greedy server selection phase. The total cost related to the network bandwidth requirements is a decreasing function of the number of servers, while the opening cost increases with the number of servers. If the opening cost is insignificant compared to the network bandwidth, the optimal solution includes all servers. The opposite situation is also possible. In Figure 12(a) the minimum is obtained for a set of 5 servers, whose location are also selected at this point. The solution does not satisfy the I/O bandwidth and storage space constraints. Figure 12(b) presents I/O capacity distribution before and after the adjustment. The requirements are lowered for all servers. However, the capacity assigned to one location is larger than for the remaining ones. This server is selected as a destination most of the time when it is involved in request redirection due to smaller increase in cost.

VII. CONCLUSIONS

We have introduced a proxy-assisted periodic broadcast system with multiple geographically distributed servers. Proxy servers cache part of each video to minimize the amount of data delivered by the servers. Given a set of video and the requests coming from a number of communities, each server chooses a mode of transmission, either unicast or broadcast to minimize the I/O bandwidth requirements. We have analyzed how distributing requests for a given video among a number of servers affects the aggregate I/O and WAN bandwidth usage. We found that WAN bandwidth usage decreases with the number of servers and strongly depends on the locations of the servers as expected. I/O bandwidth requirement on the other hand, reaches its optimal value in a single server system. The I/O bandwidth function shows more complex behavior than that of WAN bandwidth. We have defined a solution to the video placement and delivery problem that explores this trade-off between WAN and I/O bandwidth requirements. Our heuristics allows to reach a desirable value of the aggregate I/O bandwidth requirement, bounded only by the optimal value while controlling WAN bandwidth usage. We are further exploring this area of research by considering video delivery in the wireless environment. Additional constraints such as bandwidth availability, power management and client mobility introduce another dimension to the problem of video streaming.

REFERENCES

[1] A. Hu, "Video-on-demand broadcasting protocols: A comprehensive study," in *Proceedings of IEEE Infocom*, vol. 1, April 2001.

[2] S. Sen, J. Rexford, and D. F. Towsley, "Proxy prefix caching for multimedia streams," in *Proceedings of IEEE Infocom*, vol. 3, march 1999, pp. 1310–1319.

[3] Y. Guo, S. Sen, and D. Towsley, "Prefix caching assisted periodic broadcast: Framework and techinques to support streaming for popular videos," in *Proceedings of IEEE International Conference on Communications*, 2002.

[4] E. Kusmierek, D. Du, and Y. Dong, "A proxy-assisted period broadcast framework for large-scale video streaming," *in submission*, 2003.

[5] S. Barnett and G. Anido, "A cost comparison of distributed and centralized approaches to video-on-demand," *IEEE Journal on Selected Areas of Communications*, vol. 14, no. 6, August 1996.

[6] R. Luling, "Static and dynamic mapping of media assets on a network of distributed multimedia information systems," in *19th Conference on Distributed Computing Systems*, 1999, pp. 253–260.

[7] C. Shahbi and F. Banaei-Kashani, "Decentralized resource management for a dsitributed continuous media server," *IEEE Transactions on Parallel and Disitributed Systems*, vol. 13, no. 11, November 2002.

[8] K. A. Hua, Y. Cai, and S. Sheu, "Patching : A multicast technique for true video-on-demand services," in *Proceedings of ACM Multimedia*, 1998, pp. 191–200.

[9] C. Aggarwal, J.L.Wolf, and P. Yu, "On optimal batching policies for video-on-demand storage servers," in *IEEE Conference on Multimedia Systems*, 1996.

[10] L. Gao and D. F. Towsley, "Supplying instantaneous video-ondemand services using controlled multicast," in *IEEE International Conference on Multimedia Computing and Systems*, vol. 2, 1999, pp. 117–121.

[11] S. Ramesh, I. Rhee, and K. Guo, "Multicast with cache (Mcache): An adaptive zero delay video-on-demand service," in *Proceedings of INFOCOM*, vol. 1, 2001, pp. 85–94.

[12] C. Aggarwal, J.L.Wolf, and P. Yu, "A permutation-based pyramid broadcasting scheme for video-on-demand systems," in *Proceedings of IEEE Conference on Multimedia Systems*, 1996.

[13] K. A. Hua and S. Sheu, "Skyscraper broadcasting: A new broadcasting scheme for metropolitan video-on-demand systems," in *Proceedings of Sigcomm*, 1997, pp. 89–100.

[14] L. Gao and D. F. Towsley, "Supplying instantaneous video-on-demand services using controlled multicast," in *ICMCS, Vol. 2*, 1999, pp. 117–121.

[15] L. Gao, Z.-L. Zhang, and D. F. Towsley, "Catching and selective catching: efficient latency reduction techniques for delivering continuous multimedia streams," in *ACM Multimedia (1)*, 1999, pp. 203–206.

[16] B. Wang, S. Sen, M. Adler, and D. Towsley, "Optimal proxy cache allocation for efficient streaming media distribution," in *Proceedings of INFOCOM*, vol. 3, 2002, pp. 1726 –1735.

[17] W. Ma and D. H. Du, "Reducing bandwidth requirement for delivering video over wide area networks with proxy server," *IEEE Transactions on Multimedia*, vol. 4, no. 4, pp. 539–550, Dec 2002.

[18] Z.-L. Zhang, Y. Wang, D. H. C. Du, and D. Shu, "Video staging: a proxy-server-based approach to end-to-end video delivery over wide-area networks," *IEEE/ACM Transactions on Networking*, vol. 8, no. 4, pp. 429–442, 2000.

[19] W. Ma and D. H. Du, "Design a progressive video caching policy for video proxy servers," in *IEEE Trans. Multimedia*, To appear.

[20] M. Baentsch, L. Baum, G. Molter, S. Rothkugel, and P. Sturm, "Enhancing the web's infrastructure: From caching to replication," *IEEE Internet Computing*, vol. 1, no. 2, pp. 18–27, 1997.

[21] M. Daskin, "A new approach to solving the vertex p-center problem to optimality: Algorithm and computational results," *Communications of the Operations Research Society of Japan*, vol. 45:9, pp. 428–436, 2000.

[22] L. Qiu, V. Padmanabhan, and G. Voelker, "On the placement of web server replicas," in *Proceedings of IEEE Infocom*, vol. 3, April 2001.

[23] J. Kangasharju, J. Roberts, and K. Ross, "Object replication strategies in content distribution networks," in *Proceedings of Web Caching and Content Distribution Workshop*, 2001.

[24] J. Almeida, J. Krueger, D. Eager, and M. Vernon, "Analysis of educational media server workloads," in *Proceedings of NOSSDAV*, 2001.

[25] M. Chesire, A. Wolmand, G. Voelker, and H. Levy, "Measurement and analysis of a streaming-media workload," in *3rd Usenix Symposium on Internet Technologies and Systems*, 2001.

[26] M. R. Garey and D. S. Johnson, *Computer and intractability. A guide to the Theory of NP-Completness*. W.H. Freeman and Company, 1979.

[27] S. Acharya, B. Smith, and P. Parnes, "Characterizing user access to videos on the world wide web," in *Proceedings of Multimedia Conferencing and Networking*, 2000.

APPENDIX

REQUEST REDIRECTION AND SERVER PREFIX SIZE

We demonstrate that redirecting requests originating in a community whose proxy selected a larger prefix for the video than the central server prefix, to a different server does not change the I/O bandwidth consumption at the server. The proxy prefix being larger than the server prefix suggests that the part of the video that a client has to receive from the server is only transmitted in the broadcast mode. Recall that the I/O bandwidth consumption for broadcast depends only on the number of channels and not on the number of clients. Therefore, redirecting some of the requests from the server does not directly affect the I/O bandwidth requirements. However, redirection does change the global popularity of the video. Thus, it may affect the server prefix selection and the I/O bandwidth requirements as a result. We will show that such changes does not occur. That is, the minimum I/O bandwidth consumption is achieved for the same server prefix size a_l^j if $b_i^j > a_l^j$.

Let i be the community whose requests for video j are redirected away from server l. Server prefix size for video j was selected in such a way that the I/O bandwidth requirements were minimized. The I/O bandwidth consumption is a sum of I/O bandwidth consumption due to unicast transmissions of the video prefix and I/O bandwidth consumption due to broadcast of the video suffix. The total is expressed as a function of the number of segments in the server prefix in the following way:

$$W_l^j(x) = |m^j - x| + \sum_{i=1}^{M} \alpha_i^j U_{il}^j \tag{10}$$

Let the proxy prefix of video j be larger than server prefix a_l^j. Then the I/O bandwidth consumption after redirecting requests from community i denoted by $\hat{W}_l^j(x)$ satisfies the following conditions:

$$\begin{array}{ll} W_l^j(x) = \hat{W}_l^j(x) & \text{if } x \le |b_i^j| \\ W_l^j(x) < \hat{W}_l^j(x) & \text{otherwise} \end{array} \tag{11}$$

In other words the two I/O bandwidth curves are identical up to $x = |b_i^j|$ since $U_{il}^j(x) = 0$ up to this point. The first difference occurs at $x = |b_i^j| + 1$, the I/O curve after redirection is lower (Figure 5(b)). We will show that $\hat{W}_l^j(|b_i^j|+1) > W_l^j(a_l^j)$ and thus, the minimum is still obtained with server prefix a_l^j.

The broadcast I/O bandwidth consumption (first component of the sum in Equation (10)) is a linear and thus strictly decreasing function of the server prefix size x [1]. We show now that the unicast consumption is a non-decreasing convex function due to the characteristics of U_{il}^j by checking the midpoint condition. Assume that $U_{il}^k(x) > 0$, i.e., the prefix size of some video k is smaller than x. Then:

$$U(\frac{1}{2}((x+2)+x) = U(x+1) = \frac{\sum_{p=1}^{x+1} S^p - b_i^k}{m^j} \tag{12}$$

where S_p^j is the size of the segment p of video k. Recall that segments sizes form a Fibonacci sequence: $S_p^j = S_{p-2}^j + S_{p-1}^j$. Therefore:

[1] Recall that $|m^j - x|$ expresses the number of video segments in the suffix.

$$\frac{1}{2}(U(x+2)+U(x)) = \frac{1}{2}\frac{(S^k_{x+2}+S^k_{x+1}+\sum_{p=1}^{x}2S^p)-2b^k_i}{m^k}$$
$$> \quad U(\tfrac{1}{2}((x+2)+x)$$

Hence, $U^k_{il}(x)$ is a convex function and due to linear character of broadcast I/O bandwidth consumption so is the total I/O bandwidth consumption. Thus:

$$\hat{W}^j_l(|a^j_l|) < \hat{W}^j_l(|b^j_i|) < \hat{W}^j_l(|b^j_i|+1) \tag{13}$$

The first inequality follows from the assumption that the proxy prefix is larger than the server prefix before redirection, while the second follows from the fact that $W^j_l(x)$ is a convex function.

When requests for video are directed *to* server whose prefix for the video is smaller than proxy prefix, the unicast I/O bandwidth consumption increases for $x > b^j_i$ and hence, the server prefix size does not change.