Interactive Technology & Smart Education (2006) 1: 31-44
© 2006 Troubador Publishing Ltd.

An efficient resource management system for a streaming media distribution network

Adrian J. Cahill and Cormac J. Sreenan

Mobile and Internet Systems Laboratory (MISL), Department of Computer Science, University College Cork, Cork, Ireland
Email: {a.cahill,cjs}@cs.ucc.ie

This paper examines the design and evaluation of a TV on Demand (TVoD) system, consisting of a globally accessible storage architecture where all TV content broadcast over a period of time is made available for streaming. The proposed architecture consists of idle Internet Service Provider (ISP) servers that can be rented and released dynamically as the client load dictates. This paper examines issues of resource management and content placement within this Video Content Distribution Network (VCDN). The existing placement algorithm is computationally expensive and in some cases, infeasible to execute within any reasonable length of time. This work proposes a number of new placement heuristics each of which attempts intelligently to reduce the search space so that only the best proxies are considered for replica placement. An extensive evaluation of these placement algorithms is carried out to identify a good placement algorithm without being computationally expensive.

Keywords: Resource Management, Video On-Demand, Digital Video Recorder, Replica Placement Algorithms

1. INTRODUCTION

Digital video recorders (DVR) such as TiVo (Tivo, 2005) are changing the way TV is being viewed. The ability to intelligently record content and provide an easy use play-back facility has meant that no longer is the viewer restricted to the TV broadcaster's schedule, but can now watch the program at anytime (after broadcast) with the addition of commercial skipping features as well as the typical VCR functionality to which most people are accustomed.

The focus of this research is to extend the abilities of DVRs by removing the necessity for local storage, and replacing it with an always-recording global storage. With such an approach, users of the system would no longer need to be sitting in front of their DVR to view a stored file, but instead would retrieve the object from a local video server over a high-speed Internet connection. In doing this, globally broadcast TV content could be available to users on-demand. There are a number of challenges that need to be overcome when designing a TV on-Demand (TVoD) system. Issues such as (i) Latency; clients will not be satisfied if the video does not begin streaming almost immediately, (ii) Reliability; the quality of the delivery stream should be consistently high, jitter and quality degradation are undesirable, (iii) Content Control; ensuring that Digital Rights Management infringements cannot occur, (iv) Resource Management; due to the size of these high-quality video files, operating and deployment costs for the system need to be monitored to ensure the system is operating as efficiently as possible. The primary focus of this research is the efficient management of resources when providing TVoD services. It is believed that resource management will become an important aspect of video delivery in the future, as the expected quality of video objects is always increasing to match improvements in client Internet connection speeds. This further increases the load on the video servers. If a TVoD system is to be deployed, then it should be capable of efficiently serving content to a similar sized user base as the existing terrestrial broadcasting model.

System resources such as server storage space, disk and network I/O have always been an important factor in determining the scalability of multimedia distribution architectures. One of the key differences with the distribution of web documents (HTML pages and images) is the relatively short time the server resources are required in the delivery process, unlike streaming large video objects which require server resources to be tied-up for much larger periods of time. Optimizing the resources required to serve a number of clients involves optimizing (i) the content placement and (ii) the number of replicas of an object. But these parameters are influenced by the client request patterns, which are constantly changing; therefore it will be necessary to frequently re-evaluate the resource usage.

In previous work by the authors (Cahill & Sreenan, 2003; Cahill & Sreenan, 2005), Content Distribution Networks (CDNs) and Peer-to-Peer networks (P2P) were considered as possible distribution architectures for a TVoD system. It was found that P2P networks typically exhibit a lack of content control and network reliability, whereas CDNs exhibit strong control and reliability, but lack the ability to dynamically grow with increased client load. The authors proposed a new hybrid CDN-P2P architecture, termed Video Content Distribution Network (VCDN). VCDN is a content distribution infrastructure over which TVoD services can be provided. Unlike typical CDNs such as Akamai (Akamai, 2005), VCDN does not operate over a private network, but rather over a shared network. Resources such as distribution servers, storage space and bandwidth are leased from suitable service providers when client load increases, and when the load abates, the resources can be released again. This new CDN model removes the high start-up costs associated with CDNs, while also being dynamic in both size and resource usage. This is particularly important when considering TV distribution, as the expected user set, and hence resource requirements typically vary diurnally (less users in the morning than evening). Internet Service Providers (ISPs) and Datacenters typically over-provision their networks to allow for future expansion. As a result of this over-provisioning, some of their servers and disk space may be lying idle. These service providers can advertise their willingness to partake in VCDN by acting as peers, for which micro-payments will be made according to resource usage. It is expected that by agreeing to partake in the CDN, servers will not leave the CDN without providing adequate notice, thus allowing the system to find an alternative source for the affected clients.

Currently, the VCDN architecture uses a computationally expensive placement algorithm, which does not scale very well. This work proposes a number of new placement heuristics each of which attempts to intelligently reduce the search space so that only potentially optimal proxies are considered for replica placement. In doing this the required execution time can be greatly reduced, while in some cases, suffering no loss of placement optimality. As well as the new placement heuristics, this work also extends VCDN with the introduction of proxy hierarchy. The proxy hierarchy can be used to aid content placement decisions by initially deciding which region of the network would best serve a client, and then only considering the proxies within this region during content placement decisions. This is described in more detail in section 3.4.

The remainder of this paper is organised as follows: Section 2 examines the plethora of related work that exists in this area. Section 3 gives an overview of the VCDN architecture and its new proposed extensions. Section 4 outlines the placement heuristic algorithms and their parameters. Section 5 describes the simulation environment and provides extensive evaluation of the proposed heuristic placement algorithms. These algorithms are evaluated to determine the most suitable algorithm for the VCDN architecture; this involves analysis both of the running time and placement efficiency of each algorithm. Section 6 provides the reader with a view of the future trends in TV viewing and multimedia entertainment, and finally, Section 7 outlines the findings and conclusions of this work.

2. RELATED WORK

The decision over where content should be placed within a network can be modelled as a variation of the well-known Facility Location Problem (Mirchandani & Francis, 1989; Arya, *et al.* 2001), which has been described as an NP-hard problem. The reason this problem is so difficult to solve is due to its inability to scale, as the number of possible outcomes to the problem is a function of the number of facilities and objects located at these facilities (proxies and replicas in the context of this paper). To overcome this issue a number of alternative approaches and heuristics have been proposed.

One proposed solution is to distribute the content within the network such that the overall distance between clients and their requested object is minimized. In Kangasharju *et al.* (2001) the authors proposed minimizing the number of Autonomous Systems traversed when clients request objects from the server. The authors compared four replicating strategies, and concluded that a single CDN server with knowledge of the entire system was capable of making the best decision as to the placement of proxies. They propose using combinatorial analysis to locate the optimal layout, which can be computationally expensive in large-scale networks such as those expected in the VCDN architecture. This problem would be amplified by the need to constantly assess the current replica placement to ensure the system is always in a resource effective state.

Li *et al.* (1999) also carried out research in this area, but the authors only considered a tree topology network. By using a tree topology the authors have limited the path between any two nodes in the network, to a single path. The Internet is not configured in this fashion, as there are a number of routes to a given destination. The authors also only deal with the instance where there is one web server, which is cached at multiple locations.

Qui *et al.* (2001), model the replica placement problem as a K-median problem. In their work they assume that each replica site should contain a complete replica of the origin server. This is not desirable for TV content, as some media objects are expected to have higher request rate in certain areas of the network than others, as would be the case with news programs for example. Again the authors use a combinatorial algorithm, which as previously mentioned is not scalable.

Probably the most relevant related work is that carried out by Nguyen *et al.* (2003). In their work the authors also propose using a shared network infrastructure as the basis for an overlay distribution network suitable for the distribution of large objects. The authors also propose a flexible network provisioning approach where resources such as storage and CPU can be leased as required. The authors provision their network based on a cost function and an expected client demand for a set of objects. As a result of provisioning their network in advance (based on this expected demand) the authors run the risk of over/under provisioning their network resulting in either resource starvation or waste. This would not be suitable for a TV distribution architecture as the variability in TV viewers is expected to be quite large between daytime hours and evening hours for an average weekday (due to potential viewers being at work/school). Therefore, particularly in a TV distribution network, content placement should be a frequently analysed to ensure the network is operating in a cost effective manner. This could be achieved by dynamically adapting content placement with changing client loads.

More recent work in the area of video streaming has been focused on the use of P2P networks as a distribution architecture. P2P networks have gained immensely in popularity over the past number of years, due to their ability to share vast quantities of content efficiently. Recently, researchers have begun examining the potential of P2P networks for large-scale video distribution. P2P networks such as KaZaa (KaZaa, 2005) and Gnutella (Gnutella, 2005) have been used to distribute large objects such as feature length movies, but they typically perform this transfer without real-time constraints such as those that would be introduced when streaming content over a P2P network. Other issues also arise if P2P networks were to be considered as a suitable means to deliver high-quality TV objects, such as insufficient client uplink capacity, clients ability to depart from the network at will and also a lack of suitable digital rights management schemes. Also, although P2P networks have shown to be able to efficiently distribute popular objects, they may not be suitable for the distribution of unpopular content, as few requests will be made for the object and as a result, few replicas will exist.

Some research exists which examines these issues such as Padmanabhan *et al.* (2002) Choon-Hoon *et al.* (2005), Guo *et al.* (2003), Courcoubetis & Antoniadis (2002) and Tran *et al.* (2003), but in general these problems still remain. In Hefeeda *et al.* (2004), for example, the authors examine the issue of network stability in a streaming P2P network and propose dividing a video object into a number of small segments and caching these on the peer nodes. A requesting peer would first identify a list of peers with the required segments and retrieve the segments as required. In the event of a serving peer departing the network, and initial buffering period is designed to mask the switching time as a peer selects a backup peer with the missing required segment.

Xu *et al.* (2003) propose a novel distribution architecture to overcome some of the limitations in both CDN and P2P networks. A hybrid architecture uses a CDN to initially serve content to the requesting clients, with each participating client then forming a peer node in a P2P network. Over time the architecture changes from CDN-only to CDN-P2P and finally purely P2P with the CDN nodes being used for indexing purposes only. In doing this, the CDN can quickly seed a P2P network, and then remove the objects thus freeing the CDN servers for new content. Although this approach might work well for popular objects, less popular objects may reside on the CDN for long periods of time (as the P2P network will not be sufficiently seeded), as a result of this the CDN could become saturated with unpopular content thus reducing resources for newer popular objects. Also, the authors do not explain how content is initially distributed among the CDN servers, which would have a particularly large effect on resource usage within a distribution network for large objects such as high-quality video.

Finally, aside from P2P and CDNs, a number of broadcasting approaches have been proposed as a means of efficiently managing resources when distributing video objects to a large client base (Aggarwal *et al.*, 1996b; Hua & Sheu, 1997; Viswanathan & Imilelinski, 1995; Hu, 2001). Broadcasting approaches typically suffer from a lack of client control, as content is sent out over multicast channels from a server and periodic intervals, as a result of this, the delay between a client making a request for an object and the time when delivery begins may be quite high. Batching, patching and other improvements have been proposed in the past (Dan & Sitaram, 1996; Eager *et al.*, 2000; Hue *et al.*, 1998; Griwodz, 2004), but these are all hindered by the lack of suitable multicast support within the Internet.

3. ARCHITECTURE DETAILS

This section provides the reader with an overview of the VCDN architecture and its core components. In designing a suitable architecture for TV distribution a number of issues needed to be considered. TV viewing patterns change diurnally, i.e. typically a higher percentage of the population view TV in the evening time than in the morning time. This can be attributed to users being in school or work. Previous work (Cahill & Sreenan, 2005) has examined the suitability of both CDNs and P2P networks to TV distribution. They were both found to be lacking in a number of areas, such as lack of content control and reliability or dynamic expansion and running expense. To overcome these issues, the authors propose a new hybrid CDN-P2P architecture, termed VCDN. This architecture consists of leased service provider resources, such as storage space, CPU power and bandwidth. With such a design, this network does not incur a large setup cost, and can dynamically alter its size and resource requirements according to current system load. In doing this, the following goals can be achieved:

- Eliminate the necessity for client side storage.
- Access previously broadcast content from any location where a high speed Internet connection is available.
- Remove the barrier of broadcast range that exists in terrestrial broadcasting.
- Eliminate high setup costs associated with CDNs.
- Efficiently manage system resources such as storage and bandwidth.

3.1 VCDN Components

Figure 1 shows a high level depiction of the VCDN components. Within the VCDN architecture, a number of VCDN-owned proxies would be deployed with TV record facilities. These proxies record live TV, digitize it and store the object along with suitable metadata. This content is now available for retrieval by system users. Clients can request objects using the search servers in either query or browse mode. Once a TV object is selected, the client is directed to the most suitable proxy, which contains a replica.

The following is a detailed list of the VCDN components and their role within the network.

Proxies
A proxy can exist in any of two forms, Idle and

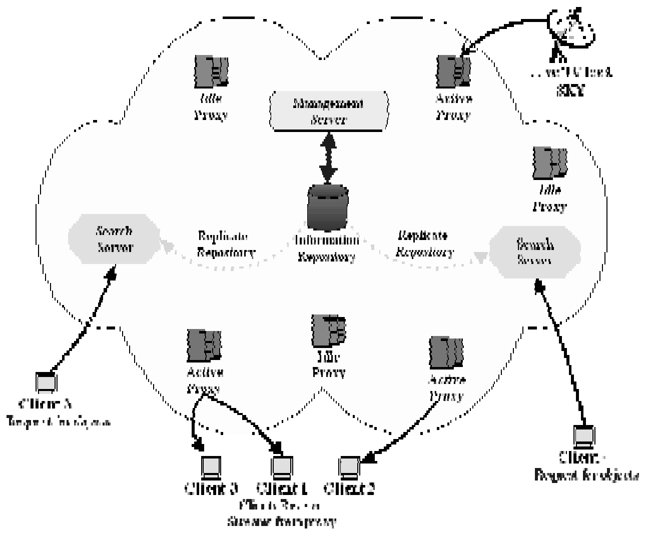

Figure 1 VCDN Component Layout

Active. An Idle proxy is a proxy that can be leased from a service provider such as ISP or a Datacenter. When Idle, a proxy cannot directly partake in the distribution of objects. Under times of increased client activity, the Management Server may decide to dynamically lease one or more of these Idle proxies for use in distributing content, at which point the proxy will change state to Active.

Clients
A client consists of any set-top box or PC that is used to retrieve content from a proxy within the network.

Cluster
A cluster is described as a group of clients that view the same content and are located in close proximity within the network. There have been a number of research papers (Hefeeda *et al.,* 2004; Amini, 2004), which examine the topic of client/request clustering ranging from simplistic techniques (grouping by Autonomous Systems) to more complex techniques such as spatial partitioning. Clustering is performed to reduce the complexity of content placement decisions, by assigning clients into clusters that are topologically close together we can infer that an optimal placement for the cluster is likely to be optimal for all the clients within the cluster (for a reasonably sized cluster).

Information Repository
The information repository maintains the state of the network, such as the location of all objects within the network, the list of active proxies, and all metadata that is associated with a video object. This metadata will be used to facilitate client searching and browsing of archived material.

Management Server
The management server is used to monitor the overall network. The tasks of the management server include gathering both statistical and accounting details and performing content control. In the event of a Pay-Per-View scheme, this server would be used to grant or revoke access to content.

Search Server
The search servers are used as an interface to the network for clients. These servers can be queried or browsed depending on the users requirements. Information from the Information Repository is cached on the search servers to aid in load distribution and to prevent the Information Repository becoming a bottleneck.

3.2 System Resources

As mentioned in the related work section, a number of techniques exist for distributing objects, though not all are concerned with resource management. Current CDNs such as that provided by Akamai primarily focus on reducing the distance between a client and the requested object. This is achieved by replicating the requested object to a CDN server located in proximity to the client. CDN providers generally are not concerned with link costs and storage costs, as they own all the required hardware. The approach outlined in this work assumes that the CDN operator does not own the resources such as the servers, disk space and high capacity links, but rather that these resources can be leased on demand (for which a fee would be negotiated earlier). Therefore, when deciding on a number or replicas of an object to make, and where these replicas will be stored, in-depth analysis of the expected resource usage is required. This decision is made even more important by the large size of high-quality video files.

This work captures the key resources involved in video distribution though the cost model could be extended to account for other resources quite easily. Future work may involve expanding this cost model to include aspects such as link quality and available capacity; this information could be used to replicate content on the opposite side of a congested link for example.

Streamed Network Cost $\alpha_{p,c}$: A cost associated with streaming video content from a video proxy ‘p’ to a cluster ‘c’. For the purposes of the cost function, this will be considered a cost per byte/per hop.

Bulk Transfer Network Cost β_{sp}: A cost associated with the inter-proxy transfer of a video object between proxy ‘s’ and proxy ‘p’, as occurs when a new replica is being created. For the purpose of this work, it will be considered a cost per byte/per hop. This is considered separately to streamed network cost as this provides a fine-grained cost model that facilitates a different pricing depending on the network requirements of the connection, for example QoS requirements.

Storage Cost ε_p: A cost associated with storing content on a proxy ‘p’. This is considered to be a cost per byte/per unit time.

Each cost outlined above could be set for each individual proxy, thereby allowing a placement algorithm to take advantage of a fine-grained cost model. This would allow certain proxies to be considered more valuable than others, for example a proxy in a large city could have a larger storage cost than that of a proxy in a rural area. In doing this, a competitive market could be formed, where ISPs lower their resource costs in a bid to entice more usage of their otherwise idle resources. The costs outlined above are all stored in a information repository and the content is pushed out to the proxies periodically. The repository can also be queried on-demand, in the event of a new proxy joining the CDN.

3.3 Assumptions

Currently, on-line storage businesses provide storage facilities by renting disk space in bulk. Our research looks at a different pricing model, where storage can be purchasing storage in this fine-grained model we could provide a more cost effective solution for video distribution. Future work will examine the effects of removing this assumption, and acquiring resources in bulk as is currently the case, for example renting server space from a Datacenter whereby they provide disk space in tens or hundred of gigabyte chunks.

3.4 Hierarchical VCDN Architecture

In previous work (Cahill & Sreenan, 2005), the VCDN architecture was proposed and some initial approaches to content placement were investigated. This work proposes a number of heuristic placement algorithms, one of which exploits a hierarchical proxy layout to determine which regions of the network are most likely to be cost effective at serving a given cluster. For example, a cluster in Ireland is unlikely to choose a proxy located in America due to the large differential in number of network hops. VCDN is divided into a number of layers as shown in Figure 2. Since the focus of this work is resource management and content placement, the Management and Cluster layers are not be covered in great detail.

The management layer is responsible for maintaining information regarding the current state of the network, the metadata associated with stored content, and all accounting and statistical information associated with each user and file.

The cluster layer comprises of clusters, where a cluster is defined as a group of clients located in close proximity which are all viewing the same

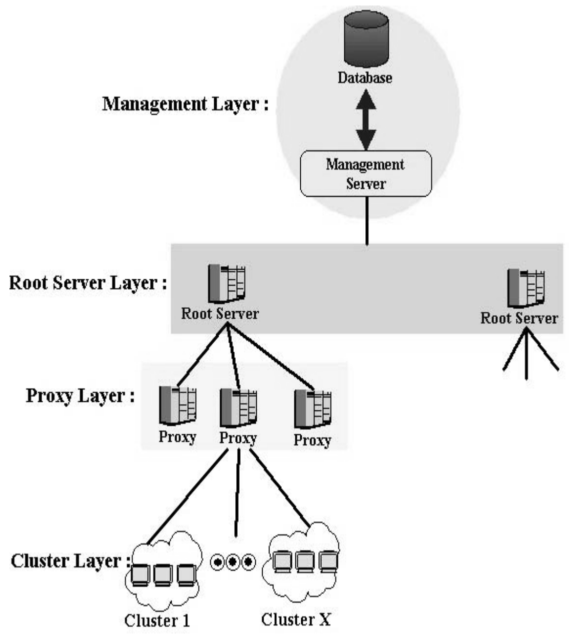

Figure 2 VCDN Component Hierarchy

content. By grouping clients that are close together in the network into a single cluster, it can be inferred that all clients within the cluster should experience similar network constraints. As a result of this, locating the optimal proxy to serve this cluster (single entity), should infer that it is the optimal proxy for all clients within the cluster.

The root server layer comprises root servers that are used to segment the network into domains. Each proxy is assigned to the domain of its closest root server. When a proxy invokes the content placement algorithm, the proxy initially determines which domain to use to serve the cluster set, and then determines which proxy within the domain. This algorithm will be described in more detail in section 4.

The original VCDN algorithm assumed global knowledge of the network, including what replicas were available on each proxy. This information is expected to change frequently and therefore would result in a lot of control messages being shared among active proxies. With the advent of a proxy hierarchy, this state information could be maintained by the root servers, thereby reducing overall network traffic and system complexity. In the event that a proxy required this information, it could query the root server on-demand.

3.5 Resource Management

Storage space and link bandwidth are the resources that are believed to most influence high-quality video distribution over a shared infrastructure. Due

Table 1 Parameters to Cost Function

Symbol	*Definition*
P_p	Proxy 'p'
$S(m)$	File size of movie m (bytes)
B_c	Bytes remaining to be served in cluster c
$\alpha_{src,\,dst}$	Bulk Delivery Cost: The cost associated with the bulk delivery of 1 byte per hop on the link between P_{src} and P_{dst}
$\beta_{src,\,dst}$	Stream Delivery Cost: The cost associated with the streaming of 1 byte per hop on the link between P_{src} and P_{dst}
ε_p	Storage Cost: Cost of storing 1 byte on Proxy P_p (per unit time)
T_c	The largest time remaining in the delivery of a stream to cluster C, i.e. time when the last client within the set of clusters will reach the end of the video stream.

to the large nature of high-quality video files, their placement within the network can have immense impact on the required resources to deliver the objects. To this end, a cost function was proposed which calculates the resource requirements for delivering content from any proxy to any cluster. This cost function can then be used to quantify the relationship between two or more content placement layouts. This cost function and its parameters are described below.

The cost of delivering an object from a proxy to a cluster is comprised of three costs: (i) Replication Cost $RC_{s,p,m}$ is the cost required to replicate movie m from P_s to P_p, and is given in Eqn. (1), (ii) Storage Cost SC_{pm} is the cost of storing movie m on P_p and is a function of the length of time storage is required for, as shown in Eqn. (2) and (iii) Delivery Cost $DC_{p,C}$ is the cost of streaming content from P_p to all C clusters, shown in Eqn. (3). The total cost of serving all C clusters from P_p is given in Eqn. (4), where the replication cost is only included if movie m is not already available on P_p.

The cost of using proxy (P_p) to serve N clients, all viewing movie *m* is given by the following equations:

$$RC_{s,p,m} = \beta_{s,p} \cdot \delta_{s,p} \cdot S(m) \qquad (1)$$

$$SC_{p,m,c} = \varepsilon_p \cdot S(m) \cdot T_C \qquad (2)$$

$$DC_{p,C} = \sum_{c=1} (\alpha_{p,c} \cdot \delta_{p,c} \cdot B_C) \qquad (3)$$

$$Cost_{s,p,m,C} = [RC_{s,p,m}] + SC_{p,m,c} + DC_{p,C} \qquad (4)$$

where B_c are the total bytes remaining to be served to cluster *c*.

4. PLACEMENT ALGORITHMS

The previous section gave the reader a brief overview of how to quantify the resource usage associated with serving a group of clusters from a given proxy. The goal of the placement algorithms outlined in this paper is to determine the resource requirements of serving a set of clusters from a set of proxies and choosing the arrangement yielding the lowest resource requirements. The placement algorithm is executed on each proxy autonomously whenever some requirement is met, such as a new client has joined this proxy. The placement algorithm selects all clusters *'C'* viewing the requested movie *'m'* and calculates the resource requirements to delivery the requested movie to each of these clusters from each proxy within the set of proxies *'P'*. In the event that a placement is found that results in less resource costs than the current placement, a placement re-shuffle occurs. To prevent oscillation occurring, where content is redirected back and forth between a set of proxies, the placement algorithm only changes placement is the resource cost associated with the new placement (taking all aspects of the current load into account) is less than the resource cost of the current placement. As a result of this, the placement algorithm can only improve placements and therefore oscillation cannot occur.

This section outlines four placement algorithms that vary in algorithm complexity and information required. These algorithms will be evaluated in section 5 to compare aspects such as overall resource usage and their execution time.

4.1 Problem Statement

As previously mentioned, the placement algorithm currently in use by the VCDN architecture does not scale well, and will be shown to require an exponential (P^C) running time, where *P* is the total number of proxies in the network and *C* is the number of clusters involved in the placement evaluation. The goal of this work is to develop a placement algorithm that can efficiently manage system resources, while also executing within acceptable bounds.

4.2 VCDN Placement Algorithm

The VCDN placement algorithm considers all proxy-cluster combinations to determine optimal placement, where the number of clusters involved in the calculation is a function of the number of clusters viewing a given movie on the proxy performing the placement evaluation. Therefore the number of calculations *'n'* performed during a placement evaluation is: n = P^C

Where P is the total number of proxies (active and idle) within VCDN, and C is the number clusters viewing the requested movie on the current proxy. As previously mentioned, this does not scale very well as values for both P and C could be quite large. The following are a number of heuristics that have been proposed to reduce the number of calculations required while also maintaining close to optimal resource management.

4.3 Best_X Algorithm

This algorithm is similar to the original VCDN algorithm described above, but performs placement evaluations on a reduced number of proxies. This algorithm takes a parameter *'X'* which equates to the number of proxies that will be considered for each cluster, i.e. if X = 1, then only 1 proxy will be considered for each cluster. Periodically, an ordered table is generated outlining the best proxies to use when delivering content to each cluster. Where best is determined using the cost function defined in Eqn. (4). If the value of X = 1, then only the 1st best proxy will be considered, with X = 2 the best 2 proxies will be considered, therefore the number of calculations *'n'* performed during a placement evaluation is: n = X^C

4.4 Hierarchical CDN Algorithm

Again, this algorithm builds from the initial VCDN algorithm, but once again attempts to reduce the size of the evaluated proxy set. This is achieved by introducing a hierarchical structure to the proxy layer as shown in Figure 2. Under this scheme, when the placement algorithm is invoked, the algorithm initially determines which root server would best serve each cluster. Once determined, the algorithm only considers the proxies within the domain of this root server. In doing this, it is hoped that the initial step will determine which region of the network is likely to contain the most suitable proxy, and when that is determined, only the proxies in that region are considered. This results in the following average number of calculations *'n'* required for placement evaluation:

$$\text{Best Case: } n = R^C + R\left(\frac{P}{R} - 1\right)^{\frac{C}{R}} \tag{5}$$

$$\text{Worst Case: } n = R^C + R\left(\frac{P}{R} - 1\right)^{C} \tag{6}$$

where R is the number of root servers in the network, P is the total number of proxies (active and idle) and C is the number of clusters involved in the evaluation. The placement algorithm consists of two separate tests. The initial test to determine which RootServer is likely to contain a suitable proxy requires R^C calculations. Whereas the second test to determine the most suitable proxy within the domain requires on average:

$$R\left(\frac{P}{R} - 1\right)^{\frac{C}{R}}$$

calculations, but could require as much:

$$R\left(\frac{P}{R} - 1\right)^{C}$$

if all clusters reside in the same domain.

4.5 Closest Proxy Algorithm

The closest proxy algorithm performs no resource calculations *'n'* when making decisions, other than when a client joins the network it determines its closest (in terms of network hops) proxy, and all objects requested must first be replicated onto this proxy, and then delivered to the clients. In this regard, there are no placement calculations performed. This algorithm was chosen to compare with the VCDN and proposed heuristic algorithms, to determine the level of resource optimization that can be gained by using intelligent placement algorithms, n=0.

5. EVALUATION

This section summarises the extensive evaluation performed on the placement algorithms described in the previous section. A number of experiments were carried out to determine: (i) the optimal number of RootServers to use in a hierarchical proxy system, (ii) a suitable number of proxies to evaluate when using the BestX placement heuristic and finally (iii) the most suitable placement algorithm for the VCDN architecture. To determine the answers to the above, both the execution time and the overall resource usage were considered, as determining the most suitable placement algorithm will involve a trade-off between a possible increase in overall resource usage in order to reduce the execution time of the placement algorithm.

5.1 Simulation Environment

To determine the performance of the proposed placement algorithms, a number of experiments were carried out. Initial tests were carried out with the ns-2 simulator, but due to its unnecessary packet-level detail, ns-2 was too inefficient and cumbersome for the required experiments. Instead, a discrete event-driven simulator was developed in C++. The simulator is supplied with a number of parameter files such as a topology layout and a client workload configuration.

Experiments are carried out using various sized Internet-like topologies generated using the BRITE topology generator (Brite, 2005). The topologies created comprised of 20, 60 and 100 proxy networks. A small set of TV objects, each 30 minutes in length were used, with the distribution of client requests among these objects following a Zipf distribution with parameter $\phi = 0.271$ as proposed by (Aggarwal *et al.*, 1996a).

Multiple client workload files were generated, each containing the interactions for 1000 clients. These workload files were generated using the multimedia synthetic workload generator MediSyn (Tang *et al.*, 2003), with client arrival rates modelled as a Poisson distribution with arrival rate λ of 0.01(100 clients per minute) as proposed by Wang *et al.*, 2004). The simulations carried out in this work use multiple workloads, each created using the same MediSyn configuration. The resultant workloads, though containing the same number of clients, will vary in requested object, location within the network and arrival time. This will be useful to examine the variability that can occur as client locations change within the network. Previous work by the authors examined the effects of varying many aspects of the client workload file including modelling users desire to channel hop (watch a TV program for a number of seconds and then change the channel in search of something more appealing). This experiment was not repeated as channel hopping is not expected to be as frequent in a TVoD system, as clients can request any object to view, and therefore is expected to request a TV program he/she will like. For a more detailed discussion regarding the effects of client workloads, and approaches to counteract the potential channel-hopping problem, the reader is directed to (Cahill & Sreenan, 2005).

5.2 HCDN Evaluation

The HCDN placement heuristic was evaluated to determine both its resource management ability shown in Figure 3 and its execution time shown in Figure 4. Since the execution time of an algorithm can vary depending on the hardware on which the algorithm is being executed, it was decided to express the execution time using the number of calculations performed by the algorithms.

As can be seen in Figure 3, all instances of the HCDN algorithm yield similar resource usage for a given workload file. The minor fluctuations occurring between instances of the HCDN placement heuristic are due to potentially good placement locations becoming unavailable as the potential proxy for a replica is currently acting as a RootServer (which do not partake in object distribution). Since RootServers are selected on the grounds of segmenting the network into equal sized domains, the RootServers in use when, for example R = 4 may not be the same as when R = 5. This accounts for the occurrences when potentially good placements are unavailable when R = 3 but become available again when R = 4.

The overall number of calculations required during these placement evaluations is shown in Figure 4. The variation in the number of calculations performed in workload 4 is an artifact of the client workload and can be attributed to the resultant change in number of clusters involved in

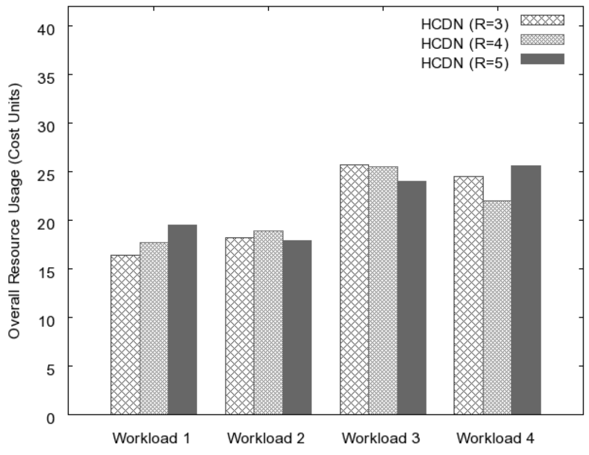

Figure 3 Resource Usage for HCDN Heuristic

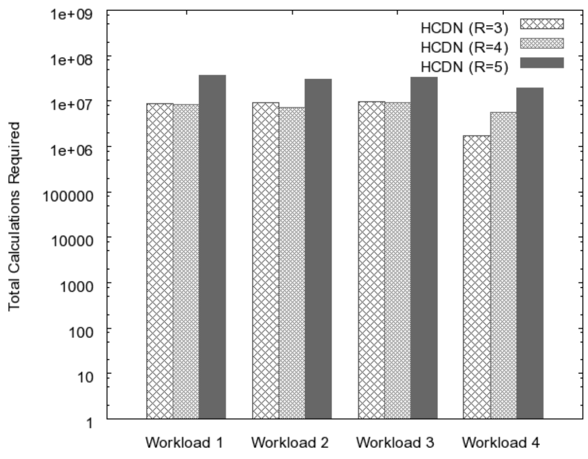

Figure 4 Required Calculation for HCDN Heuristic

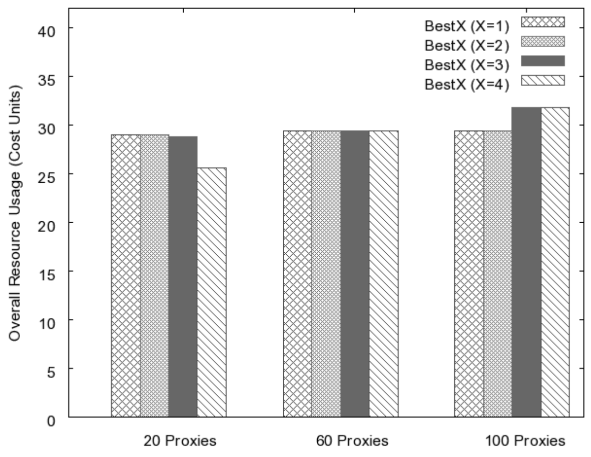

Figure 5 Resource Usage for BestX Heuristic

content placement decisions, variable *'C'* in Eqns. 5 and 6. For example, the average number of clusters involved in a placement evaluation is less when using R = 3 than when R = 4 for the given workload. As the number of RootServers *'R'* increases, the number of calculations required drops, this occurs as $R\,(\frac{P}{R}-1)^{\frac{C}{R}}$ is the dominant factor in determining the number of calculations. As R increases, a point will be reached when R^C becomes the dominant factor in determining the number of calculations required, this can be seen quite clearly in Figure 4. The values when *'R'* was less than 3 and greater than 5 were not shown in the graphs, as they offered no extra information.

5.3 BestX Evaluation

The BestX heuristic takes a parameter *'X'*, which is the number of proxies to consider as potential placements for each cluster. By increasing the value for *'X'* and hence increasing the number of proxies considered, the execution time for the algorithm increases also. This can be seen from the graphs in Figure 6, the number of calculations required when X = 1 is approximately equal to the number of clients (as the placement algorithm is executed at least once when each client joins), but as the value of *'X'* increases, the number of calculations and hence execution time, rises immensely. Therefore to reduce the execution time of a placement algorithm, its essential to reduce the number of proxies or the number of clusters which are to be considered. It was found that varying the number of proxies considered did not have any major impact on the resource management of this algorithm, as shown in Figure 5. This indicates that only considering the best proxy for each cluster is sufficient in most cases, and in some cases can actually yield better resource management than considering multiple proxies, as there is a reduced likelihood of choosing a sub optimal proxy.

It was also found that when using a small topology, there are many opportunities where the same proxy can serve multiple clusters. In such circumstances, the algorithm can identify instances when a single proxy can serve multiple clusters, thereby resulting in savings in storage cost and replication cost. Unfortunately, the number of calculations required also increases the number of clusters involved in any particular placement evaluation. As the topology size grows, the average number of clusters involved in a placement evaluation decreases, this can be attributed to the increased number of possibilities for a clusters placement, hence the likelihood of clusters being served by the same proxy is greatly reduced. This can be seen in Figure 7.

Figure 6 shows the difference in execution time as the number of examined proxies increase. It can be seen that the execution times linearly increase,

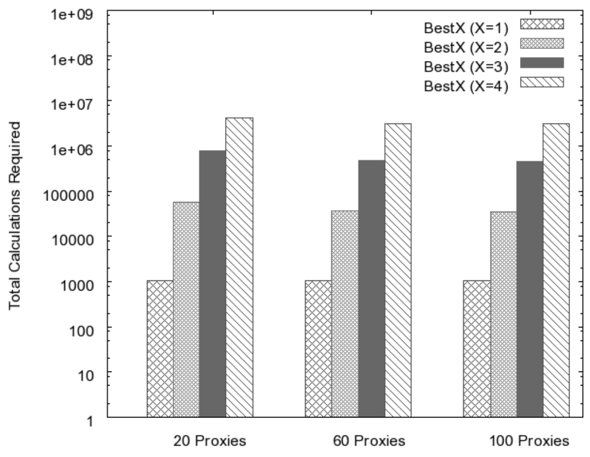

Figure 6 Required Calculations for BestX Heuristic

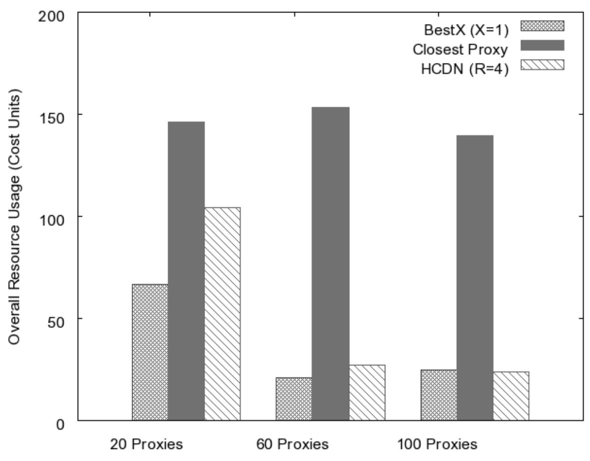

Figure 7 Resource Usage of Heuristics

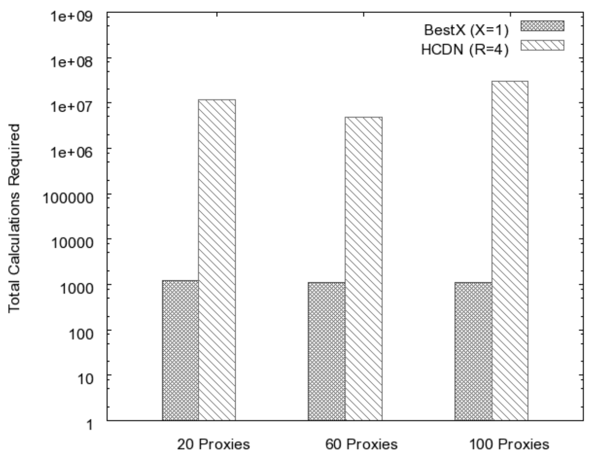

Figure 8 Required Calculations for Heuristics

and that the number of calculations is independent to the number of proxies in the topology. Given, that these placement algorithms are expected to be executed frequently, it is vital that their running time be minimized.

5.4 Resource Management

This section looks at the overall resource management abilities of the proposed placement heuristics. It was not possible to evaluate the VCDN placement algorithm under any of the proposed topologies, as the execution time required was too large. Figure 7 shows the overall resource usage when placing replicas using the BestX, Closest Proxy and HCDN algorithms. As can be seen, using the BestX algorithms yields minimal resource usage in most cases. The increase in resource usage for the BestX placement algorithm is attributed to the fact that the algorithm only considers one proxy as a possible placement, and in the event of this proxy reaching its capacity of clients, then the cluster(s) will remain being served by the currently more costly server. This did not occur in any of the larger topology networks, as the clusters were spread out over a greater number of proxies.

As expected, the Closest Proxy algorithm performed poorest, as it always selected the proxy closest to the clients to serve the content, regardless of whether they are more expensive than further away proxies. Similar results are shown for larger topologies as shown in Figure 7.

As shown in Figure 7, for any reasonably sized topology (60 proxies and 100 proxies), both placement heuristics yielded almost identical resource usage, but the number of calculations required while using the HCDN algorithm was significantly more than that required by the BestX algorithm. Also, although it is not very apparent from the graphs, the number of calculations required by the HCDN heuristic is more than six times larger than that required by the same workload using a smaller topology, as shown in Figure 8. Again, this was expected, as the number of proxies within each domain has increased significantly. For this reason, even though HCDN can significantly reduce the number of evaluated proxies during a placement evaluation, it still requires too many evaluations (the number of calculations required exponentially grows with topology size) to be considered useful. The current implementation of the HCDN algorithm assumes a single level of hierarchy.

By increasing this to a multi-level hierarchy, it would be possible to reduce the number of calculations further, but it is unlikely to perform as good as the BestX algorithm which in its current state with X = 1, has the minimum possible execution time. Also, the BestX algorithm has yielded better placements on almost all occasions when compared with the HCDN heuristic, therefore no advantage could be gained by using such an approach.

6. FUTURE TRENDS

It is believed that due to the immense financial investment required to setup a CDN, using shared infrastructure and resources will become a more common approach to content distribution, for example, the IETF Content Distribution Internetworking (CDI) Working Group is examining the possibility of CDNs working together to serve clients at times of high loads. In such an environment, suitable middleware will be required to control the infrastructure, providing features such as the ability to securely add and remove content dynamically from leased servers, and also resource monitoring for all servers partaking in the network.

As mentioned earlier, the costs for each resource can be set individually, this facilitates a bidding nature when assigning a value to a resource. If the value is too high, then the VCDN placement algorithm will select a cheaper resource, and the expensive resources will remain idle and not earning any money for the service provider. Issues pertaining to the payment of the service providers, either in the form of micro-payments or some other suitable approach needs to be examined. These would be very interesting research areas, which need to be examined before any content distribution architecture could be deployed over a shared infrastructure.

The future of conventional broadcast TV is believed to be currently lying in the balance. More and more users are choosing to view their TV programs with DVRs such as TiVo, where they can view the program at a time that suits them and without the annoyances of advertisements. Content creators, may soon decide to skip the TV broadcaster and distribute their content directly to the users, thereby earning more profits by cutting out the middleman. In such an environment, distribution systems such as VCDN will be required.

Finally, though the work presented in this paper focuses on TV content, with some minor alterations the proposed architecture could be used for the efficient distribution of any type of objects. One proposed extension is the provision of a complete home entertainment system to the user. In such an environment, a set-top box located in the home could be used to access a globally networked music collection, or use the set-top box to play games online, where the source code required for the next level is downloaded directly to the set-top box on demand.

7. CONCLUSIONS

Content distribution has become very important in the past number of years, with increasing last-hop connection speeds, creating greater demand for large objects such as high-quality video. This work extends the previously proposed Video Content Distribution Network (VCDN). VCDN was designed to be an efficient distribution network, suitable for the delivery of high-quality video objects such as TV content. The scalability of the VCDN architecture was limited by the length of time required to perform content placement evaluations.

This work proposes two new placement heuristics suitable for the VCDN system. The heuristics attempt to reduce the examined set of proxies, intelligently removing proxies that are unlikely to be able to efficiently deliver content to a set of clients. In doing this, the number of calculations performed and hence execution time for the algorithm decreases. The HCDN placement heuristic arranges the set of proxies into a hierarchy. The placement algorithm then performs two tasks, initially it determines which region of the network is likely to contain a good proxy placement, and then determine the best proxy within that region. The BestX placement heuristic on the other-hand, proposes that an ordered list of good proxy locations is kept for each cluster. A parameter, 'X' is passed to the algorithm that determines how many proxies it should consider during placement evaluations.

Finally, an extensive evaluation of these placement heuristics was performed, outlining their resource management ability and execution time. It was shown that a direct correlation exists between the algorithm execution time and the number of proxies it considers during placement evaluation. The evaluation stage initially determined the optimal configuration parameters for both placement heuristics, in terms of the HCDN algorithm; this was to determine the optimal number of regions to divide the network into, and in terms of the BestX algorithms, it was to determine the best number of proxies to consider for each cluster during placement evaluation. Once determined, these two algorithms were compared, to identify which, if any, would be suitable for use in the VCDN. It was shown that for any reasonably sized topology, both placement heuristics yielded almost identical resource usage, but that the number of calculations required when using the HCDN algorithm was significantly more than that required by the BestX algorithm.

In summary, a placement algorithm which uses a cost model to determine the best proxy for each cluster, and then only considers that proxy as a potential location for replicas requested by the cluster, yields very good resource usage and evaluates in minimal time. Future work will examine the effect of using a coarse grained cost model, where resources cannot be purchased on a per byte scale, but rather in predefined chunks as is currently the granularity that service providers operate in.

ACKNOWLEDGMENTS

The authors wish to thank AT&T Labs Research (USA) for part-sponsoring this project. In addition, support from the EU FP6 E-NEXT Network of Excellence contributed greatly to this work, by enabling close interaction with other E-NEXT partners working on related research.

REFERENCES

Aggarwal, C.C., Wolf, J.L. and Yu, P.S. (1996a) On optimal batching policies for video-on-demand storage server. In *Proc. from the International Conference on Multimedia Computing and Systems*, Hiroshima, Japan, pp 253–258

Aggarwal, C.C., Wolf, J.L. and Yu, P.S. (1996b) A permutation-based pyramid broadcasting scheme for video-on-demand systems. *Proc. from IEEE International Conference on Multimedia Computing and Systems '96,* Hiroshima, Japan.

Akamai (2005) http://www.akamai.com.

Amini, L. (2004) *Models and Algorithms for Resource Management in Distributed Computing Cooperatives*. PhD thesis, Columbia University.

Arya, V., Garg, N., Khanderekar, R., Munagala, K. and Pandit, V. (2001) Local search heuristic for k-median and facility location problems. *Proc. from ACM Symposium on Theory of Computing*, Crete, Greece, pp 21–29.

BRITE: Boston university Representative Internet Topology gEnerator (2005) http://www.cs.bu.edu/brite/.

Cahill, A.J. and Sreenan, C.J. (2003) VCDN: A content distribution network for high quality video distribution. *Proc. from Information Technology & Telecommunications,* Letterkenny, Ireland.

Cahill, A.J. and Sreenan, C.J. (2005) An efficient cdn placement algorithm for high-quality TV content. *Proc. from Internet and Multimedia Systems and Applications - EuroIMSA*, Grindelwald, Switzerland.

Choon-Hoong, D., Nutanong, S. and Buyya, R. (2005) Peer-to-Peer Networks for Content Sharing. In: *Peer-to-Peer Computing: Evolution of a Disruptive Technology*, pp 28–65. Idea Group Publisher, Hershey, PA.

Courcoubetis, C. and Antoniadis, P. (2002) Market models for p2p content distribution. *International Workshop on Agents and Peer-to-Peer Computing, AP2PC* 2002, Bologna, Italy, July, 2002.

Dan, A. and Sitaram, D. (1996) A generalized interval caching policy for mixed interactive and long video environments. *Proc. from IS & T SPIE Multimedia Computing and Networking Conference*, San Jose, CA.

Eager, D., Vernon, M. and Zahorjan, J. (2000) Bandwidth skimming: A technique for cost-effective video-on-demand. *Proc. IS&T/SPIE Conf. on Multimedia Computing and Networking*, San Jose, CA, USA.

Gnutella: P2P Application (2005) http://www.gnutella.com.

Griwodz, C. (2004) The use of stream merging mechanisms in a hierarchical cdn. *Proc. from IS&TSPIE Multimedia Computing and Networking (MMCN)*, Santa Clara, California.

Guo, Y., Suh, K., Kurose, J. and Towsley, D. (2003) P2cast: Peer-to-peer patching scheme for vod service. *Proc. of 12th International World Wide Web Conference (WWW 2003),* Budapest, Hungary.

Hefeeda, M.M., Bhargava, B.K. and Yau, D.K.Y. (2004) A hybrid architecture for cost-effective on-demand media streaming. *Computer Networks*, 44(3): 353–382.

Hu, A. (2001) Video-on-demand broadcasting protocols: a comprehensive study. *Proc. INFOCOM*, pages 508–517.

Hua, K.A. and Sheu, S. (1997) Skyscraper broadcasting: A new broadcasting scheme for metropolitan video-on-demand systems. *Proc. from ACM SIGCOMM*, pages 89–100, Cannes, France.

Hue, K., Cai, Y. and Sheu, S. (1998) Patching: A multicast technique for true video-on-demand services. *Proc. of 6th ACM International Multimedia Conference*, Bristol, UK. ACM Multimedia '98.

Kangasharju, J., Roberts, J., and Ross, K. (2001) Object replication strategies in content distribution networks. *Proc. of WCW'01: Web Caching and Content Distribution Workshop*, Boston, MA, USA.

KaZaa: P2P Application (2005). http://www.kazaa.com.

Li, B., Golin, M., Italiano, F., Deng, X. and Sohraby, K. (1999) On the optimal placement of web proxies in the internet. *Proc. of INFOCOM*, New York, NY, USA.

Mirchandani, P.B. and Francis, R.L. (1989) Discrete Location Theory. In: *The Uncapacitated Facility Location Problem*, pp. 120–168. John Wiley, New York.

Nguyen, T.V., Chou, C.T. and Boustead, P. (2003) Provisioning content distribution networks over shared infrastructure. *Proc. from 11th IEEE Internation Conference On Networks (ICON)*, Sydney, Australia.

Padmanabhan, V., Wang, H., Chou, P. and Sripanidkulchai, K. (2002) Distributing streaming media content using cooperative networking. *Proc. of Workshop on Network and Operating System Support for Digital Audio and Video,* Miami, FL, USA. NOSSDAV.

Qiu, L., Padmanabhan, V. N. and Voelker, G. M. (2001) On the placement of web server replicas. *Proc. from IEEE INFOCOM,* pages 1587–1596, Anchorage, Alaska.

Tang, W., Fu, Y., Cherkasova, L. and Vahdat, A. (2003) Medisyn: A synthetic streaming media service workload generator. In *Proc. of Workshop on Network and Operating System Support for Digital Audio and Video*, Monterey, California, USA. NOSSDAV.

Tivo (2005). http://www.tivo.com/.

Tran, D. A., Hua, K. A. and Do, T. (2003) Zigzag: An efficient peer-to-peer scheme for media streaming. *Proc. of the 22nd IEEE INFOCOM*, San Francisco, CA.

Viswanathan, S. and Imilelinski, T. (1995) Pyramid broadcasting for video on demand service. *Proc. from SPIE Multimedia Computing and Networking*, pp. 66–77, San Jose, CA.

Wang, B., Sen, S., Adler, M. and Towsley, D. (2004) Optimal proxy cache allocation for efficient streaming media distribution. *IEEE Transaction on Multimedia, Special Issue on Streaming Media*, 6.

Xu, D., Chai, H.-K., Rosenberg, C. and Kulkarni, S. (2003) Analysis of a hybrid architecture for cost-effective streaming media distribution. *Proceedings of SPIE/ACM Conference on Multimedia Computing and Networking (MMCN 2003)*, Santa Clara, CA.

Adrian J. Cahill received his BSc degree in Computer Science at the University College Cork Ireland in 2001 at which point he begun his PhD Studies. Adrian is part of the Mobile and Internet Systems Laboratory (MISL) at the University. His research interests include multimedia networking, content distribution, and distributed systems. Further details at: http://www.cs.ucc.ie/misl

Cormac J. Sreenan is Professor of Computer Science at University College Cork in Ireland. Previously he was a researcher at AT&T Labs Research, and at Bell Labs Research in Murray Hill, NJ, USA. He holds a PhD in Computer Science from the University of Cambridge. His research interests include mobile and multimedia networking. He is currently on the editorial board for ACM/Springer Multimedia Systems Journal and has recently served as guest editor for IEEE Journals and Magazines. He is a Fellow of the British Computer Society. Further details at: http://www.cs.ucc.ie/~cjs