VCDN: A CONTENT DISTRIBUTION NETWORK FOR HIGH QUALITY VIDEO DISTRIBUTION

Adrian J. Cahill and Cormac J. Sreenan

Department of Computer Science
University College Cork
Cork, Ireland
{cahill,cjs}@cs.ucc.ie

1. ABSTRACT

Content Distribution Networks (CDNs) have aided web servers in dealing with hot spots by replicating popular content on servers close to the clients. Typical CDNs are static in nature and as such, do not be able to react well with changing client viewing trends. This paper outlines the design of a Video Content Distribution Network (VCDN), which unlike tradition CDNs, is designed to automatically replicate and/or move content on to proxies located closer to the clients generating the requests. This paper outlines the costs involved in the system and the proposed algorithm for content placement and replication. The placement algorithm monitors client locations and server load on an ongoing basis, and alters the set of serving proxies and/or content location to minimize the overall resources being used. As a result of this, the VCDN is expected to yield near optimal cost effective video distribution.

2. INTRODUCTION

This paper describes a method of using the Internet as both a transmission and storage medium for TV content. By doing this, it is expected that clients will receive a far greater number of TV channels than when using traditional TV broadcast, while also minimizing the risk of missing a program which they may have desired to watch. By storing the content in a globally accessible location, it is hoped that client local storage can be reduced, if not eliminated, while also providing a level of content management which is not possible by using local storage in the home.

There have been issues in the past with web servers becoming overloaded due to a large number of client requests for web documents (eg text and images) over a small period of time. In an effort to overcome this problem, the services of CDNs have been employed. CDN's such as Akamai [1] replicate popular content on a number of strategically placed servers within the Internet, thus providing multiple access points for clients which results in lower latencies while also providing load balancing for the web servers. Currently the task of replicating content on a CDN is often done manually by a system administrator, and as a result the content distribution on the CDN does not change very often. This problem is amplified tremendously when dealing with the delivery of continuous media such as TV content, due to the fact that TV content is much larger in size than typical web documents, and requires more resources to deliver. This work describes the possibility of using a CDN like system to aide in the delivery of video content. For this approach to work efficiently, it will be necessary to improve on the current CDN approach by designing a system which can automatically replicate or move content to an area of the network which would best serve the clients.

The Video Content Distribution Network (VCDN) proposed consists of a two tier architecture. On the lowest layer is the proxy infrastructure, consisting of video servers (proxies) which can be made active or dormant (dormant proxies incurr no resources). The VCDN will also have a feature to allow Internet Service Providers (ISPs) the ability to allow idle servers to join the VCDN dynamically, acting as a video proxy. For this service a fee would be paid to the respective ISP wich would be negotiated beforehand. The upper layer of the VCDN consists of the control software, whose job is to monitor the resource costs incurred by the VCDN and where necessary alter the proxy placement and/or content location, to reduce the resource cost.

The remainder of this paper is organized as follows. Section 3, outlines some of the related work examined. Section 4, presents the system architecture indicating what components are involved and their roles. Section 5 addresses the problem of moving and replicating content within the network. Section **??** outlines the tasks of the individual components within the system and finally Section 6 shows the authors' conclusions and future plans for this project.

3. RELATED WORK

3.1. Content Placement

The problem of how many proxies to use, and where to position content, is very similar to the *Facility Location Problem* [2], which has been described as an NP-hard problem.
Kangasharju *et al.* [3] tried to solve the problem by 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.
Li *et al.*[4] 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.
Clarke *et al.* [5] describe the placement and replication strategy used in their implementation of Freenet [6], which is a decentralized peer-to-peer (p2p) style system. Within Freenet, requests for content travel between nodes within the system, with the idea that each hop gets you one step closer to the requested content. Once found, content returns to the node which instigated the request via the search path (path the search request followed), with each node along that path caching a copy of the content. If another request arrives at one of these nodes within a short period of time, then the node should be able to serve that new request. Over time, as fewer requests reach the original source node, and many reach the *new* source node, the content on the original node is removed to make way for newer more popular data. As a result of these steps content has *moved* from one location on the network to another location which is closer to the requesting clients.

3.2. Content Delivery

There are a number of inherent drawbacks with the existing TV delivery approaches (Satellite, Cable and Terrestrial), such as limited broadcast range and limited number of channels available. As previously mentioned, the Internet could be used as a distribution network eliminating the problems incurred by using the conventional approaches, but the architecture and resources are not fully in place yet to facilitate this. Many clients still connect to the Internet using a dial-up connection which yields very low data transfer rates, though as broadband technology becomes more widely distributed and cost effective, more and more video content will be available on the Internet.

Clients can view on-line multimedia content in one of two ways, *streaming* or *downloading (file transfer)*. Streaming content involves transferring the content from the source (server) while viewing the content as it is being received. This is significantly different to the *file transfer* model associated with *downloading* content from the web, where all the content must be received before playback or usage of the file can begin. As a result the start-up latency experienced by a client for *file transfer* is quite large. In an effort to improve this start-up latency, Pyramid broadcasting [7] was proposed. The authors proposed dividing the content into increasing size blocks, clients would then download the blocks in sequence, initiating playback after the first block has been fully transferred. These methods of delivering content can be used to deliver content to a small number of clients over unicast links, but the limited I/O resources will prevent web servers from delivering content to many clients using unicast technology. In order to combat this problem, multicast techniques can be used. These range from *batching* [8, 9, 10], *patching* [11] and *chaining* [12] for streaming content, while *pyramid broadcasting, skyscraper broadcasting* [13] could be used for the file transfer method of delivery.

An important aspect of the VCDN is the delivery of content from the servers to the client set-top box. This section outlined some of the existing content delivery approaches which could be taken.

4. ARCHITECTURE DETAILS

In this section, the architecture behind the VCDN is described, while also showing how this system can provide an improvement on the current continuous media delivery system. By placing video proxies/servers throughout the Internet, each with the ability to store and replay content, a network storage facility could in theory be provided for all video content which is being broadcast throughout the world. Clients would then be permitted to retrieve content from the most suitable video proxy which contains the desired content. In doing this, the following goals can be achieved:

- Eliminating the necessity for clients to store content and hence reducing the need for client disk-space.

- Minimize the possibility of clients sharing content, without the content creators consent.

- Reduce the need for clients to explicitly or implicitly schedule a recording for a program which he/she will not be able to view.

- Provide a far greater choice of content to the clients, as any content which is stored within the network could be available to the clients as long as certain requirements have been met (e.g. some content may be pay per view).

This method will remove the restriction on content transmission such as TV broadcast range, and deliver the content to all clients connected to the Internet. It is assumed that clients wishing to view live content can do so without having to burden the video servers, but rather by joining the relevant multicast group and viewing the content as it is streamed to their set-top box. Clients who wish to view this content at a later stage, may do so by contacting the relevant proxies in an on-demand fashion.

In order to find content within this network it is proposed that the client set-top boxes have a web interface whereby the user can search for some stored content, or possibly look up an on-line TV guide. The results of which are in the form of a Universal Resource Indentifier (URI), which when selected is mapped to the most suitable video server which contains the content. This mapping will be described in more detail in a later section.

The architecture described so far does not limit this technology to video content alone. Any type of object could be stored on this network, and transferred to the clients set-top boxes. In fact these set-top boxes could perform the tasks of many entertainment devices found in a typical home, depending on the content type which is delivered to it. It is reasonable to think that entire music collections may be contained within this network, providing the clients with an always up-to-date jukebox. Games could also be available for download, thus turning your set-top box into a games console, allowing you to compete with other users on-line. This could all be controlled using the same pay-per-view scheme as that which will be applied to some if not all TV content.

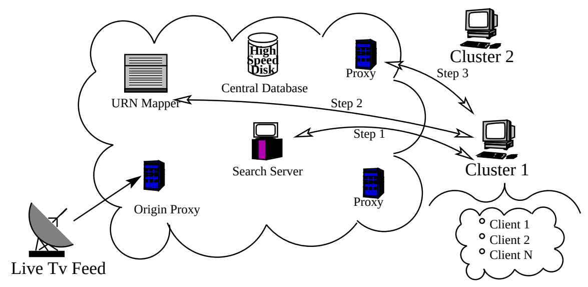

Figure 1: VCDN components

4.1. Network Model

Fig. 1 depicts the components and the steps required to retreive content from the VCDN. The tasks of the individual components are outlined below.

Media Object A media object is a stored media file which can be requested for viewing.

Clients Clients are individuals with PCs or set-top boxes, which request a particular piece of content stored somewhere in the VCDN. The VCDN deals with clusters of clients, rather than individual clients. Clusters are described as groups of clients in a similar region of the network viewing the same content. By using clusters it is reasonable to make the assumption that all clients within that cluster should experience similar latency and bandwidth constraints.

Central Database The central database contains information regarding the content stored on all proxies, along with information regarding all the content stored within the VCDN. This information could be used in a search query to identify a particular piece of content which is being requested. In an effort to reduce the load on this database, it was decided to cache information regarding the contents of all proxies on the *URN mapper*, and cache a copy of the *content information* on the search servers. This should minimize the amount of network traffic required to locate a particular piece of content. To keep the cached information always up to date, the central database would be expected to send update request to the relevant servers as changes occur.

Proxy Proxies have the ability to store and deliver content. Proxies share content among themselves, and autonomously decide whether or not to move or replicate content to another proxy. Whenever content is moved or replicated within this system, an update message needs to be sent to the *central database* to reflect the new location of the content. When an *origin proxy* receives new content via its live feed from a TV broadcast stream, a different message is sent to the database. This message needs to include meta-data (such as actors names, episode details and possibly a brief description of the content) regarding the content which could be used to uniquely identify the content using a search query.

ISPs will be able to dynamically add idle servers into the VCDN for use in content storage and delivery. The details of how this is carried out has not been examined fully yet. Along with this, proxies themselves will be able to join or leave the “pool of serving proxies” dynamically. Since there will be a fee for the use of a server, these proxies have been designed with the ability to *turn off* when they are no longer of use. Proxies would later become active again if it was selected to deliver content to a cluster.

Cluster Load When clients request an object from a proxy, there is a certain load placed on the VCDN. This is a function of the demand from the cluster (number of bytes requested), and the number of hops between the cluster and the serving proxy.

URN Mapper The task of the URN mapper is to receive a URI from the client which represents the content which they are requesting. The URN mapper then looks up the local cached copy of the *content location* database table, to find which proxies contain a replica of the content. If there are multiple proxies which have replicas of the content, the URN mapper must determine which proxy is most suitable for the cluster. The URN mapper then constructs a Universal Resource Locator (URL) which will identify the desired content on the most suitable proxy based on some other heuristic.

Search Server The search server is the clients interface to the VCDN, this will provide both a search facility aswell as an on-line TV guide. The search servers are expected to return a URI for a particular piece of content, this URI will uniquely identify a piece of content with the VCDN, not an individual replica of the content. These search servers could also be accessed via existing web search engines such as google [14] allowing web searches to link to video content stored within the network.

The search server takes a query from the client and searches a local cache of the *content information* database table which it received from the *central database*. The search server then returns a list of possible results to the client, information about the content and the URI is sent to the client also. The client selects the URI which is most likely to represent the content desired, the client then contacts the URN mapper.

4.2. Costs Involved in the VCDN

To ensure that the system is always in a (resource) cost effective state, costs are assigned to various elements of the system, which will be used to influence how and where content is placed within the network. The following section outlines the elements which are believed to be of influence when deciding where proxy content should be placed.

Network Cost δ This cost is a function of the number of hops along the link between the proxy and the client, and the number of bytes which are to be delivered to the cluster. By reducing the number of hops traversed between a client and a proxy it would lower the overall network cost. The number of hops it is assumed, can be calculated using utilities such as traceroute.

Service Cost β A service cost will be associated with using a proxy, for the purpose of this paper it will be taken to be a “cost per byte” served. This fee is expected to be to an ISP for the use of one of its servers.

Transfer Cost α There would be a certain resource cost associated with transferring a media object between proxies. This cost is a function of the size of the content and the number of hops the content is to be sent over.

Running Cost λ This is a cost per unit time that a proxy is running (even if no data is being delivered), the proxy resources are still “promised” to the VCDN and for this a fee would need to be paid. This is also an incentive to *turn off* a proxy as soon as it is no longer delivering content.

The costs outlined above are all stored in the *central database* and the content is *pushed* out to the proxies periodically, or this database could be queried in the event of a new proxy joining the VCDN.

It is worth mentioning some of the extreme cases, as they further show the usefulness of the cost definitions. If *network* issues were not important then there is no motivation to locate the object close to the cluster. On the other hand, if there was no *running cost* then content could be placed on all servers and there would be no incentive to making proxies *active* or *dormant*.

5. PLACEMENT ALGORITHM

The placement of content within any CDN is of great importance to the overall efficiency of the network. Content should be placed in a location which yields the lowest resource cost when delivering the content to its clients. One of the largest costs in such a system is that of *network cost*, as this influences many aspects of content delivery including latency experienced by the client. For this reason, the VCDN has been designed to automatically react with changing client trends, and dynamically move or replicate content accordingly. Using the cost variables mentioned in the previous section, a cost function was designed as a means of locating the optimal proxy (yields lowest resource cost) to deliver content to a set of clusters.

The cost of an active proxy, (P_i) serving content to N clusters is given by:

$$J_i = \lambda(t) + \sum_{n=1}^{N} ((\beta_i + \delta_{i,n})T_n) \quad (1)$$

where T_n is the total bytes remaining to be served in cluster C_n, and $\lambda(t)$ is the cost of keeping the proxy running for 't' *time units*, where 't' is the minimum length of time the proxy needs to be running in order to serve all clients.

The cost of a activating a dormant proxy (P_r) to serve content to N clusters is given by:

$$J_r = \alpha_{i,r} + \lambda(t) + \sum_{n=1}^{N} ((\beta_r + \delta_{r,n})T_n) \quad (2)$$

where $\alpha_{i,r}$ is the cost of transferring the file from proxy P_i to proxy P_r

Thus it is more resource efficient to deliver the content to all clusters from proxy P_r instead of proxy P_i if

$$J_r < J_i$$

T_n can be measured or estimated by taking each client, and calculating how much time remains in the *stored* content being requested.

5.1. Dynamic Content Placement

The previous section shows how two proxies can be compared to determine which proxy would use the least amount of resources to deliver content to a set of clusters. Up to now, the use of a single proxy to serve a number of clusters has only been discussed, this may not yield the best results. There will be times when it would be more cost effective to deliver the same content from different areas of the network, ie serve American clients from a server in America, while serving European clients from a European server. In order to achieve this, a placement algorithm has been designed which should be able to recognize when this situation occurs, and automatically replicate or move the content to suitable servers. It is vital that this algorithm is efficient as it is expected to be executed regularly, and has to accumulate many distributed variables.

In an effort to make this algorithm execution more efficient we pre-calculate as much of the equation as possible beforehand. At each proxy, a table is constructed containing the costs of delivering **1 MB** of data from each proxy to each region of the network (fig: 2). Using simple multiplication it is possible to transform this value to reflect the cost of transferring the full content size. All information needed to create this table is available from the *Central Database*, which pushes the information out to all proxies periodically. The contents of this database can also be queried at any stage by an individual proxy in the event of a new proxy joining the VCDN.

The *number of hops* between each proxy and region of the network, will need to be available on the central database for dissemination purposes. It is proposed that each proxy calculate the number of hops to all regions of the network (using applications such as traceroute), and forward the results to the central database. Since this information is a function of the network stability, it is proposed that this task be repeated periodically. This time period is expected to be in the order of a number of days.

	$Proxy1$	$Proxy2$	$Proxy3$	...
Region 1	3	9	3	...
Region 2	7	5	5	...
Region 3	5	5	8	...
Region 4	11	3	2	...
...				

Figure 2: Example cost table for Proxy 3, showing the costs of serving 1 MB of data from each proxy to each region of the network

Each proxy would be expected to independently decide if they should move/replicate their content to another proxy. The decision on which proxy the content should be moved / replicated to could be calculated quickly by looking up the proxy *cost table*. There are three possible outcomes to this event:

No Change No other proxy can deliver the content to the cluster using less resources, in this case the currently serving proxy carries on serving the content.

Replicating Content A subset of clusters would be better served from a different proxy (this must take the cost of transferring the content between proxies). In this case the content is replicated to each of these proxies and the relevant clusters are re-directed to the new "closer" proxies. Issues of replication are not within the scope of this paper and will be dealt with in a later paper.

Moving Content All clusters are best served by different single proxy, in which case the content is moved to the new proxy and all clusters are redirected to the new serving proxy.

Content may also be replicated in the event that the load on a proxy becomes too high. In this case, the cost table is examined to locate the next most cost effective arrangement for the content. If at any stage the number of clients being served by a cluster reaches zero, then the proxy is “turned off”.

6. CONCLUSION AND FUTURE WORK

This paper outlines the limiting factors in the existing approaches to TV distribution and proposes using the Internet to overcome these limitations. This paper expands on the existing approaches of CDN’s by designing an algorithm to dynamically and automatically shift content to regions of the network which would yield better resource management. A cost function is shown which can be used to locate the proxy set which will yield the lowest resource costs when serving a set of clusters. Although the approach described within this paper only deals with one media object, this approach can be used for multiple media objects as each would be treated separately, and as described within this paper.

Future work will look at the architecture of VCDN in more detail, focusing on proxy hand-off, which occurs as clusters are re-directed to different proxies. There are many other issues which need to be examined such as delivery techniques, system requirements, bandwidth issues, how ISPs advertise their willingness to partake in the VCDN, and the effects of new proxies joining the VCDN. Future work will also include simulating how efficiently this system performs in comparison with existing CDN technologies and Video delivery mechanisms [15, 16]. This paper should be treated as a work in progress, and outlines the current state of the project.

7. ACKNOWLEDGEMENTS

The authors would like to thank AT&T Labs-Research for supporting this project.

8. REFERENCES

[1] “Akamai.” http://www.akamai.com.

[2] V. Arya, N. Garg, R. Khanderekar, K. Munagala, and V. Pandit, “Local search heuristic for k-median and facility location problems,” in *ACM Symposium on Theory of Computing*, pp. 21–29, 2001.

[3] J. Kangasharju and J. Roberts and K. Ross, “Object Replication Strategies in Content Distribution Networks.” Proc. of WCW’01: Web Caching and Content Distribution Workshop, 2001.

[4] B. Li and M. Golin and F. Italiano and X. Deng and K. Sohraby, “On the Optimal Placement of Web Proxies in the Internet.” Proc. of INFOCOM, 1999.

[5] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, “Freenet: A distributed anonymous information storage and retrieval system,” *Lecture Notes in Computer Science*, vol. 2009, pp. 46+, 2001.

[6] “Freenet project.” http://www.freenetproject.org.

[7] K. A. Hua and S. Sheu, “Pyramid broadcasting for video on demand service,” in *Proc. from SPIE Multimedia Computing and Networking*, (San Jose, CA), pp. 66–77, 1995.

[8] A. Dan, D. Sitaram, and P. Shahabuddin, “Scheduling policies for an on-demand video server with batching,” in *Proc. from ACM Multimedia*, pp. 15–23, 1994.

[9] C. C. Aggarwal, J. L. Wolf, and P. S. Yu, “On optimal batching policies for video-on-demand storage server,” in *Proc. of the International Conference on Multimedia Computing and Systems*, (Hiroshima, Japan), pp. 253–258, 1996.

[10] A. Dan, D. Sitaram, and P. Shahabuddin, “Dynamic batching policies for an on-demand video server,” in *IEEE Multimedia Systems*, pp. 112–121, 1996.

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

[12] S. Sheu, K. A. Hua, and W. Tavanapong, “Chaining: A generalized batching technique for video-on-demand systems,” in *Proc. Multimedia Computing and Systems*, (Ottawa, ON, Canada), pp. 110–117, 1997.

[13] K. A. Hua and S. Sheu, “Skyscraper broadcasting: A new broadcasting scheme for metropolitan video-on-demand systems,” in *ACM SIGCOMM*, pp. 89–100, 1997.

[14] “Google search engine.” http://www.google.com.

[15] A. Basso, C. Cranor, R. Gopalakrishnan, M. Green, C. Kalmanek, D. Shur, S. Sibal, C. Sreenan, and J. van der Merwe, “PRISM: An IP-based architecture for broadband access to TV and other streaming media,” in *Proc. of Workshop on Network and Operating System Support for Digital Audio and Video*, (Chapel Hill, NC), NOSSDAV, June 2000.

[16] S. Acharya and B. Smith, “Middleman: A video caching proxy server,” in *Proc. of Workshop on Network and Operating System Support for Digital Audio and Video*, (Chapel Hill, NC), NOSSDAV, June 2000.