TITLE: **OPTIMAL STREAMING PROTOCOL FOR VIDEO-on-DEMAND USING CLIENT'S RESIDUAL BANDWIDTH**

THEME: **INTERNET BASED COMMUNICATIONS**

Guided By: **R.Ashok Kumar,** email: professorashok@gmail.com

Presented By,

**SUBRAMANYAM.S,** email: shastri9999@gmail.com

**PUNEETH BHAT.A.H,** email: puneeth.bhat@gmail.com

**MADHUSUDAN.C.S,** email**:** madhusudancs@gmail.com

**SANTOSH G VATTAM,** email: vattam.santosh@gmail.com

INSTITUTE: **B.M.S COLLEGE OF ENGINEERING,**
**POST BOX NUMBER 1908, BULL TEMPLE ROAD,**
**BANGALORE – 560019**

EMAIL: principal@bmsce.ac.in

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

# OPTIMAL STREAMING PROTOCOL FOR VoD USING CLIENTS' RESIDUAL BANDWIDTH

## 1. Introduction:

With internet reaching out to a global audience today, people are expecting and extracting a lot of new features and facilities from it, IPTV being one of them. IPTV provides a lot of features that conventional cable television fails to provide, such as high definition video quality and VCR functions like pause, fast forward, rewind, etc. Another important aspect that makes IPTV so popular and better than conventional cable television is Video On Demand (VoD). VoD involves, streaming video files from a remote server to the client systems on an individual basis. This involves streaming of videos requested by the clients specifically to them. This provides a certain kind of customized viewing option that is absent in regular cable TV. The client requests for a particular video on a VoD server and the request is processed and served by the server.

Through VoD, the video can be viewed as it is being downloaded from the server. Along with this the VCR functions provide an extra edge over conventional television, letting the user decide when and how to watch the video. IPTV is very popular in the western world with as many as 25+ service providers providing service in different parts of the USA. Some of the major stake holders being Verizon and AT&T serving as many as 20 states between them. In India, MTNL is the only service provider currently.

The implementation of VoD requires a high bandwidth network framework consisting of a server that has sufficient storage capacity to store video files. The following table in *Fig.1* shows some of the standard video formats used in VoD[*Wiki-1*] :

| **Format Type** | **Format Name** | **Developed By** | **Bit Rate** (*Kilobits/sec*) | **Size of the Video with *10* minutes playback** (*Megabyte*) |
|---|---|---|---|---|
| .rm | Real Media | Real Networks | 350 | 50 |
| .mov | Quick Time | Apple | 1,000 | 200 |
| .avi | Audio Video Interleave | Microsoft | 1,024 | 100 |
| MPEG-2 (HDTV) | Moving Picture Experts Group - 2 | Moving Picture Experts Group | 20,480 | 1,536 |
| MPEG-4: (HDTV) | Moving Picture Experts Group - 4 | Moving Picture Experts Group | 8,192 | 600 |

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

*Figure 1*: Standard File formats used in VOD and their parameters.

IPTV generally uses HDTV videos and hence the servers need to have large storage capacities to store the video files. This constitutes one of the main requirements of a VoD network. Using MPEG standards the bandwidth required is 2133 Mbps. Maintaining such a high bandwidth requires a high bandwidth network with optic fibre channels. This makes the service expensive and impractical to implement from the client's side. To overcome these drawbacks the network is implemented in three levels. Level I has the main server with an optic fibre channel having a bandwidth in the range of 2 Gbps. At the Level II there is a cluster of Proxy servers that are connected to the server at one end and to the clients at the other. The connection on the server side is an optic fibre channel and that on the client side is a low bandwidth network that provides a bandwidth of 2 Mbps. The Level III has the set of clients that are served by the proxy and in turn by the server.

When the client requests for a video from the proxy that serves the client. This proxy in turn requests the server for the particular video. The server then streams the video to the proxy and this is further streamed to the client.

Whenever a video that is already being streamed to some client is requested the server streams it again, thereby streaming the same video to different clients and using twice the bandwidth for the same video file. The server bandwidth is of great importance in the network and the streaming of the same video twice hinders the optimal utilisation of the server bandwidth. On the other hand the uplink bandwidth on the clients' side is under-utilised. Providing a protocol for the optimal streaming of the videos using the clients' residual (uplink) bandwidth and hence reducing the load on the client constitutes our objective with this paper.

## 2. Objective:

In this paper we present a protocol to use the clients' residual bandwidth and reduce the load on the server. This is achieved by a process called "Chaining" [*Chain-2*]. We will prove that, by applying the proposed protocol the following results are obtained, during simulations:

- Optimal Bandwidth Utilisation
  - The number of streams does not increase with increase in the number of client requests
  - Network traffic is drastically reduced in the Proxy- to -Server network
  - Effective use of the previously under-utilised Client uplink bandwidth
- Cost effective streaming

## 3. Review of Literature:

The VOD streaming has attracted quite a few researchers; some of the previous works

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

have been reviewed here. There have been two kinds of works in this regard. They are:

- Protocol and architecture related works
- Building of Streaming servers

Since we are proposing a protocol we intend to concentrate more on the Protocol and Architecture related works. There have been quite a few innovative techniques and protocols proposed in this field. The Prefix Caching [*Prefix-3*] is one of them**.** Prefix caching concentrates more on the storage problems related to proxy servers, but in the current era where memory is cheap and easily available the significance of prefix caching lessens.

There is one other proposed method – "Optimized Distributed Delivery of Continuous-Media Documents over Unreliable Communication Links" [*Optim-4*]. In this method a single multimedia file is split up and sent in parts from different servers. Effectively this is like multiple servers serving multiple clients. The drawback in this system is that it is very expensive to maintain multiple servers and the server bandwidth is of great value.

In the Hybrid Model [*Hybrid-5*] proposed the model used is that multiple clients – called peers - lend or contribute their resources to the network and serve their peers. The model optimally uses all the network resources and provides very good service but the major drawback is that the model is too complex to implement. The protocol requires the service of each and every peer and hence the service becomes less reliable.

One of the recent publications by CBC-Radio Canada [*CBC-6*] proposes a Peer Assisted Content Distribution System (pCDN). According to it pCDN provides the most viable CDN option. The fundamental idea behind pCDN is that clients also called peers are not only the consumers but also the contributors for the network. In addition the Service provider will also have few Seed Servers to induce new content into the system and also to provide the content to new peers when all the peers who currently have the requested content are unavailable. At the centre of the operations, A Client requests a Multimedia content to the Service Provider's Seed server. However seed server, instead of streaming the Content, it redirects the request to other peers who already have this content. The client starts receiving this content from a subset of peers thereby decreasing the probability of not getting the content in case of connection failures associated with single server systems.

To give a thrust to this idea Warner Brothers recently made a deal with BitTorrent to sell more than 200 films over BitTorrent's[*WB-7*] P2P network. But there are several research challenges to be overcome in order to employ pCDN in a commercially viable multimedia content distribution system. The protocol proposed in this paper overcomes all the drawbacks and limitations of the above mentioned protocols.

## 4. Proposed Protocol:

The structure of the network assumed in our protocol is as shown in *Fig. 2*. The network architecture consists of a Main Server at the top of the hierarchy. This is followed by a cluster of Proxy servers that act as an interface between the clients and the Server. Each Proxy can be thought of as a regional server that serves a set of clients that belong to a particular region. One of the aspects of having proxy servers is that the probability of two clients

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

requesting the same video is more in a given region. At the same time, the more popular the video, the more the probability of a client requesting the same video. Hence the number of clients requesting for the video will be more. Thus if the Server satisfies each and every request for the same video, the server bandwidth is not optimally utilised. In order to avoid this, these popular videos may be stored in the Proxy buffer and streamed as and when the client requests for it. Since the popularity is region dependent, each proxy buffer contains the videos that are popular in a given region and the videos can be streamed directly.

*Figure 2*: Network architecture

The videos are stored dynamically, that is if the popularity of a particular video drops, the video is automatically deleted from the proxy buffer giving way to some other more popular video. The popularity is based on the number of hits that occur for a particular video. The above mentioned parameters constitute the essential requirements for the protocol.

The optimal utilisation of the server bandwidth is implemented using “Chaining”. When a video is requested by a client, the request is redirected to another client which already contains the video in its buffer, instead of being served by the server or the proxy. This process is called “Chaining”. This reduces the load on the proxy and the main server as well as puts the unused client bandwidth to good use. in order to achieve this a protocol is necessary, which constitutes the subject of the next section.

**4.1 Protocol**:

*Fig. 3* shows the protocol followed in this implementation. When a connection is established between the client and the server, the client sends a “HELO” message to the server. When the server receives this “HELO” message, it requests the client for the “MovieID”. In response the client sends the Movie ID to the server. On receiving the Movie ID, the server sends a "REDY" message to the client and waits for client's response. The client sends an “OK” message to the server indicating that it is ready to receive the movie. Once the server receives the "OK" message the server starts streaming the movie, segment by segment. For each segment the client receives, it sends an acknowledgement (“OK” message) to the server. The server sends the next segment only after receiving an acknowledgement from the client.

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

When the server is streaming a movie to Client-i, suppose Client-j requests for the same movie, then the request of Client-j is forwarded to Client-i. Now the Client-i uses its up-link bandwidth to stream the same movie to Client-j, thus taking the load off the server.

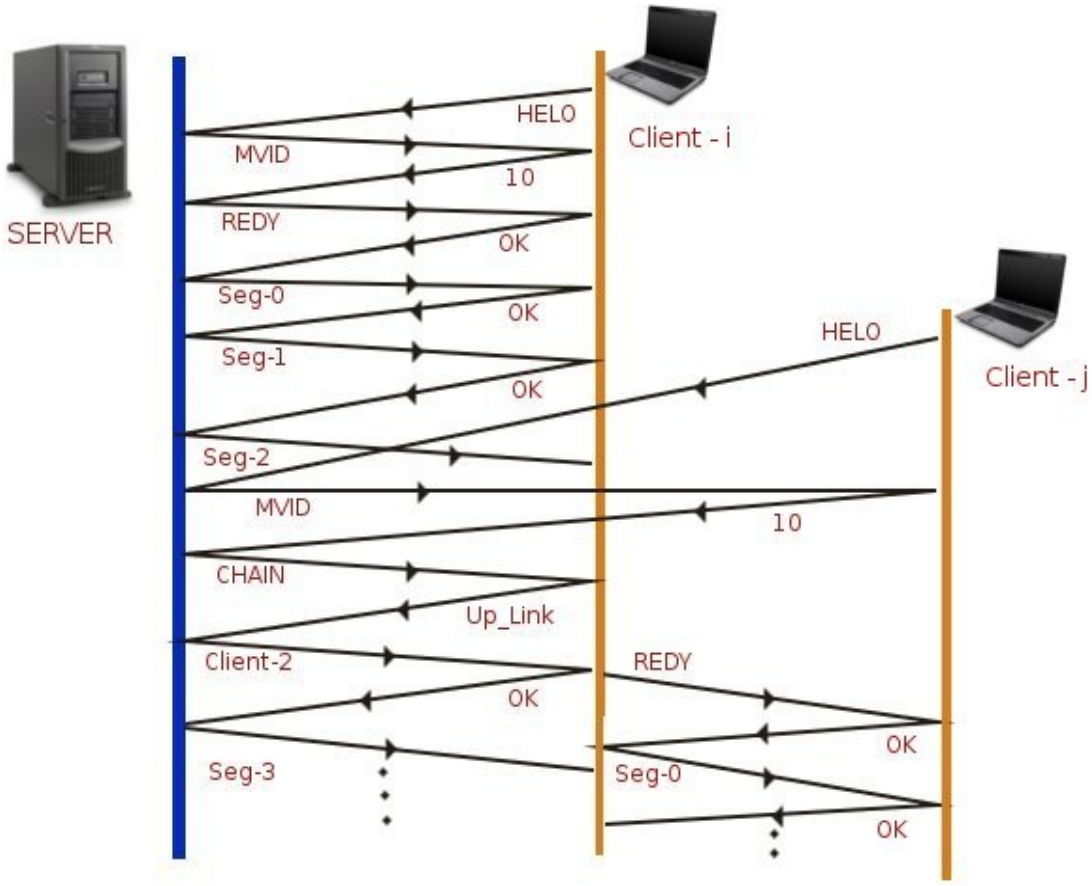



*Figure 3*: Protocol

**4.2 Algorithms:**

The algorithms that are required to implement the above protocol are as follows:

CHAINING ALGORITHM

Nomenclature:

**C**= {$C_1$, $C_2$, $C_3$, $C_4$.....$C_m$}, Active clients

**P**= {$P_1$, $P_2$, $P_3$, $P_4$.....$P_n$}, Active Proxies

$C_i$ →Client address

$P_i$ →Proxy address

$B_i$ →Buffer size used in client $C_i$

b= {($\mu_1$, $\theta_1$), ($\mu_2$, $\theta_2$)... ($\mu_i$, $\theta_i$)}, b is a list containing $\mu_i$ and $\theta_i$ where $\mu_i$ is Movieid , $\theta_i$ is popularity.

**D** $_{\{i, j\}}$ →Distance between Client $C_i$ and $C_j$

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

$\mu$→ MovieId (Request Id)
$\eta_i$→ Uplink of client $C_i$
**R** = { $r_1$, $r_2$, $r_3$……$r_k$}, A list containing current streaming request.
Bi→ Proxy $P_i$'s current Bandwidth
$\beta$→Maximum allowable buffer size of proxy P
$S_r$→ Requested segment
$S_i$→ Segment number i
**Q**→A circular queue in client holding received segments

*Proxy*

Upon receiving request from client $C_i$

```
1       if μ in R then
2               find j such that (cost[i] → D {i, j}/ ηi) < D {i, p} / Bi, where rj→μ
3               If ∃ some j such that Cj ϵ C
4                       chain client Ci to Cj
5                       θi →θi+1, where ri→ μ
6       else
7               if μ in b then
8                       initiate new stream ri to client Ci, R=R U {ri}
9                       θi →θi+1, where ri→ μ
10              else
11                      send request rk→ μ to server
12                      receive segments from server
13                      if maxsize{b} > β
14                              replace (μk, θk) in b with (μl, θl) where θk< θj ∀ j
```

The above mentioned algorithm is the proxy server algorithm. The Proxy server waits for movie requests from the clients. Whenever a request is received, the proxy checks if the movie requested is being streamed to any client. If so the "cost of chaining" is calculated and compared with that of streaming without chaining and the most economical method is chosen. If Chaining is feasible, the client is redirected to the client that has the video in its buffer. In case chaining is not a feasible option then proxy buffer is checked for the movie requested. If found the movie is transmitted by allocating a new stream to the client. In case the movie is not found in the proxy buffer, the request is redirected to the main server.

*Client*

Send a request $r_i$ to proxy $P_j$

Upon receiving SIG_READY from $P_j$

```
1       Receive segment So from Pi,Q= Q U {So}
2       cptr=0,i=0
3       if Bi < size(Si)
4               Q= Q - {Sin}
5       Q= Q U {Si}
6        cptr →cptr+1
```

Upon receiving SIG_CHAINTO

```
7               If Sr in Q
8                       stream the segments Sr to Sn
9               else
```

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

10 reject

The client requests for a video from the proxy server. If the proxy is ready to stream, the client receives an acknowledgement “SIG_READY” message. Then the movie is streamed from the proxy, segment by segment, to the client. In case the movie is being streamed to some other client, the address of that client is sent from the proxy in the “SIG_CHAINTO” message, and the stream is transmitted from the peer client. When a client receives a Movie ID and an address from the proxy, then the client starts streaming the movie in its buffer to the address mentioned, thereby contributing to the chain. If the requested movie is not in the buffer then the request is simply rejected.

**5. Findings and Analysis:**

The above mentioned algorithms were implemented using **Python** programming language for simulation. The simulation was done with the real world parameters in consideration. The following parameters were used in the simulation:

- Total Bandwidth of the server.
- Current Bandwidth utilisation
- Uplink Bandwidth of the client
- Buffer size of the client
- Popularity of the video requested
- Cost of streaming

The simulation was carried out for 10 minutes, with a single server, 5 proxy servers and 80 clients. The server had 20 movies each of size 4.9 MB. The proxy buffer was large enough to hold 50% of the data on the server .i.e. effectively 10 movies. The server to proxy bandwidth was assumed to be 2 Gbps and the proxy to client bandwidth was assumed to be 2Mbps. The uplink bandwidth of the client is assumed to be 256kbps.

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

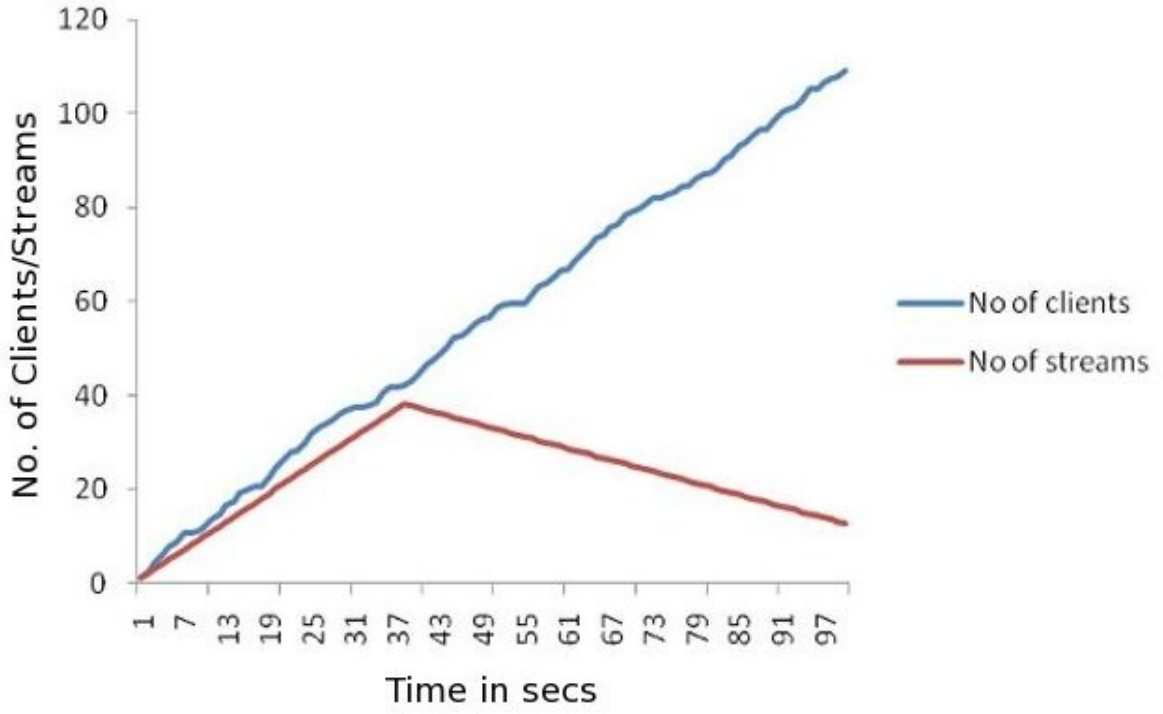



*Figure 4*: Number of clients and Number of Streams v/s time

The graph shown in *Fig. 4* is the combined graph of "Number of clients v/s Time" and the "Number of streams v/s Time". The blue line indicates the number of clients v/s time whereas the red line indicates the number of streams v/s time. We can observe that as time increases the number of clients increase. But as the number clients increases we can observe that the number of streams increases initially and then decreases. This is because the proxy or the server serves the clients only till all the popular videos have been cached in at least one place. Later on the videos are chained from these places. This point is the peak at the value 38 in the number of streams v/s time graph. After this point the number of streams decreases even though the number of clients increases.

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

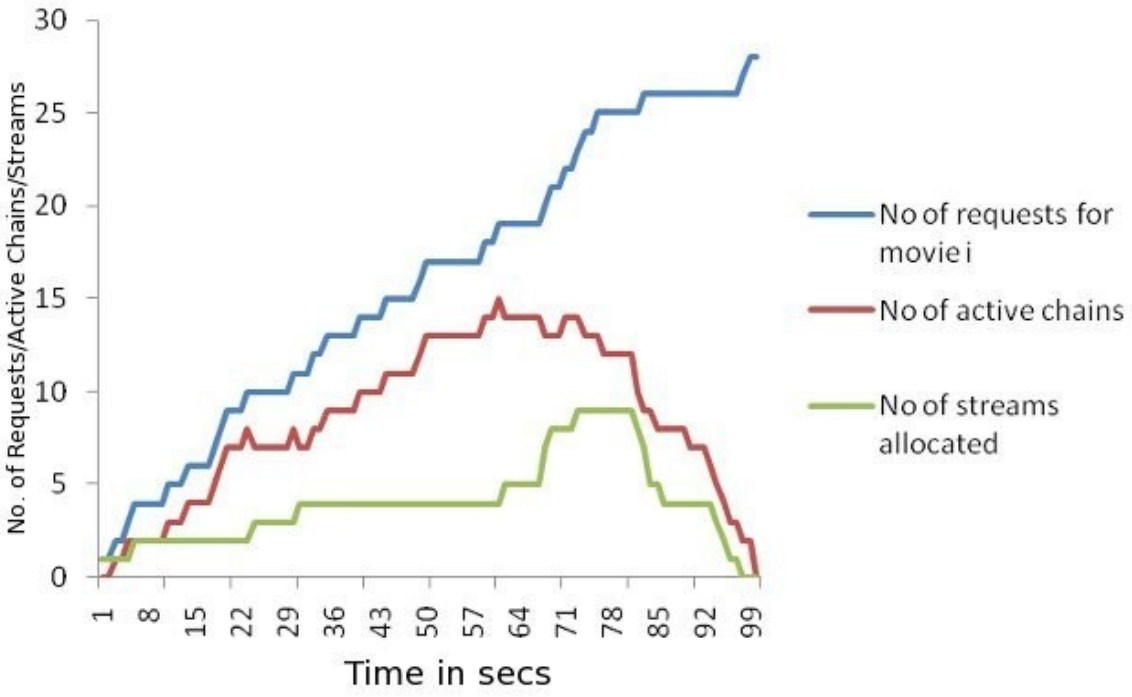



*Figure 5*: Number of streams and Number of Active chains w.r.t. time

The graph in *Fig.* 5 shows the number of streams and numberof active chains with respect to time. In this case the video requested is very popular and hence with time the number of requests for the video is increasing as the blue line in the graph rightly points out. We can observe that as the number of requests increases, the number of active chains also increases initially as shown by the red line in the graph. But at one stage when the number of active chains is around 15 it reaches a peak and then starts descending. This can be explained as follows.

Initially the video is present in very few clients, who are scattered in different locations. Hence the more number of active chains have to be created in order to serve the requests. But as the requests are served, the number of clients that have the video will grow. Once more clients possess the video, the number of active chains gradually decreases. The same is the case with the streams allocated for the video in the server as shown by the green line in the graph. But the number of streams is always less than the number of active chains ( as seen in the graph, the no . of streams is always below the number of active chains ) because of the chaining. The chaining reduces the number of streams from the server and hence the server bandwidth is conserved.

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

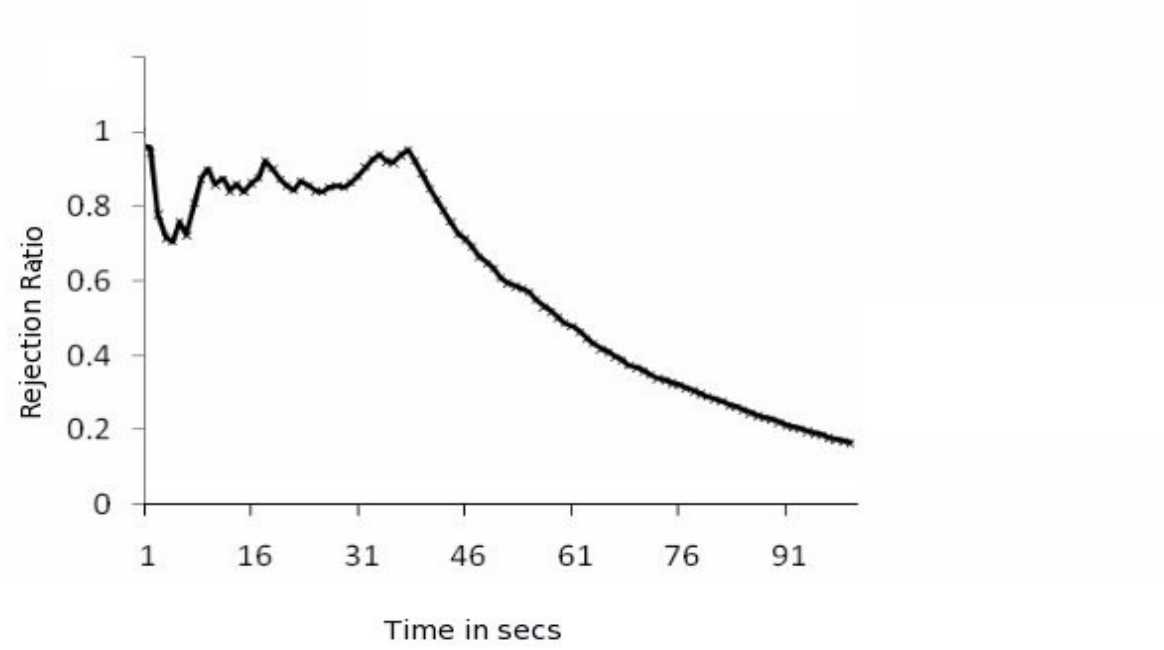
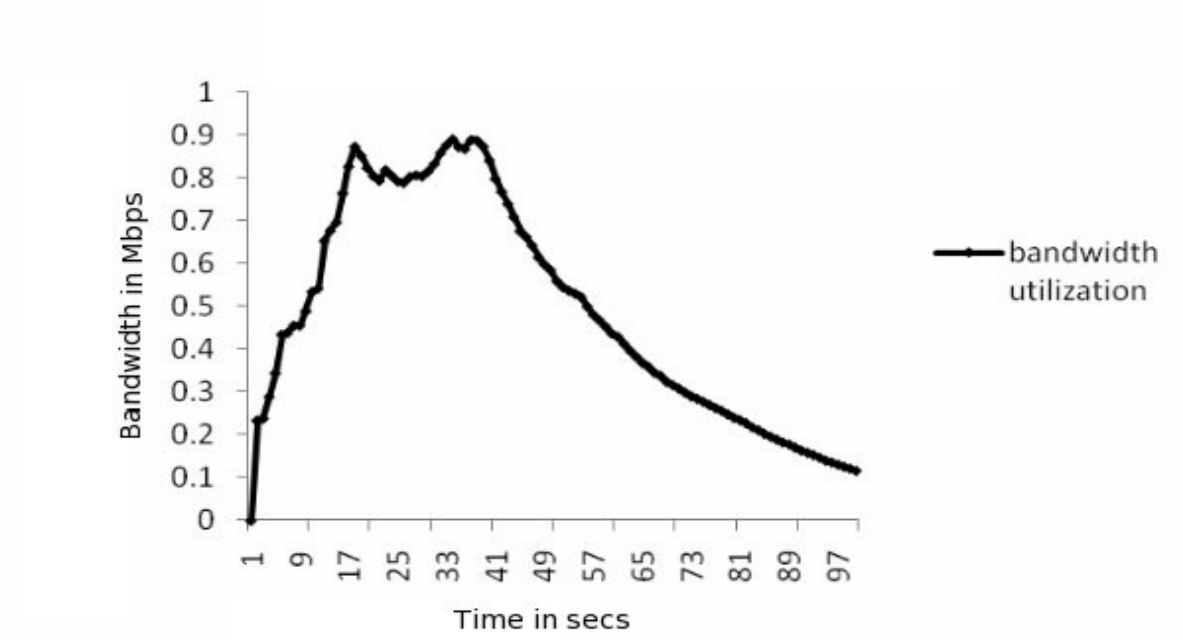



*Figure 6*: Rejection ratio with time

The graph in *Fig.* 6 shows Rejection Ratio with respect to time. The rejection ratio is defined as the ratio of the number of rejected requests to the number of active chains. It represents the fact that the client requests are rejected even though there is a possibility of chaining. This happens because even though the chaining is possible the cost of chaining exceeds the actual cost of streaming. The ratio is around 0.9 initially since the numberof clients possessing the video is small and scattered. But as the number of clients that possess the video increase the rejection ratio comes down rapidly as shown. This happens as the number of chains that serve a request grows.




This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

*Figure 7*: Bandwidth utilization with time

We can observe that the bandwidth initially increases with time and then peaks at a point and then reduces gradually to some constant value. Initially the number of requests that arrive have to be served by the server itself because the number of clients that possess the video is small, as shown in the previous graph of the rejection ratio. But once the chaining process is activated the bandwidth used of the server is reduced and hence the burden on the server is reduced to a great extent.

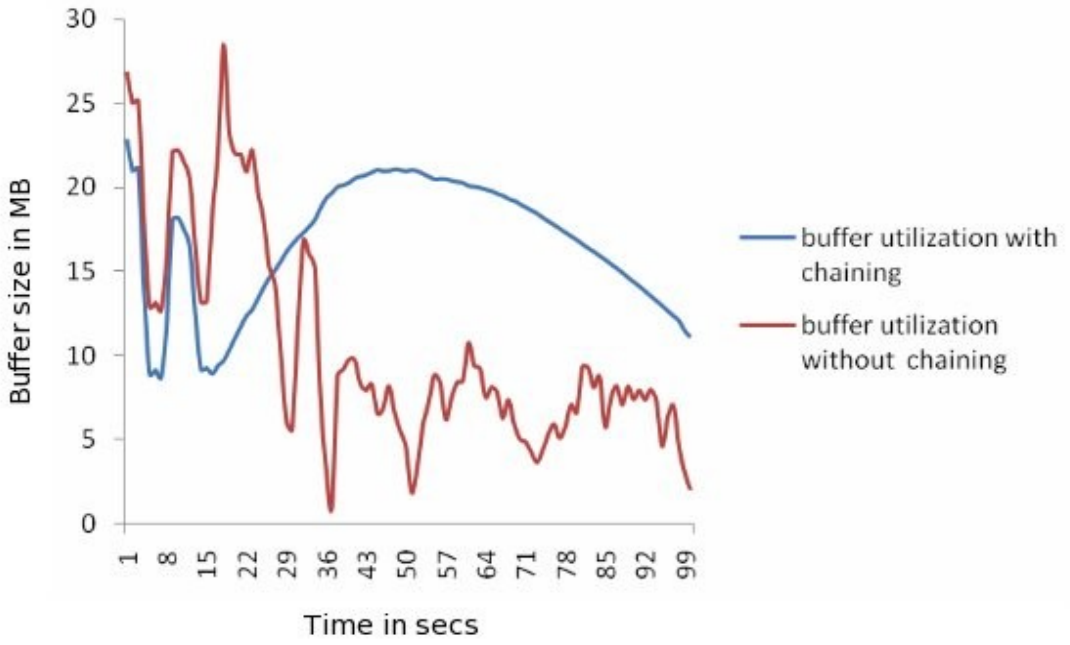



*Figure 8*: Buffer utilization

The buffer utilization graph shown in *Fig.* 8 shows the client buffer utilized. Without chaining we can observe that the buffer utilization is erratic and decreases drastically with time. But with chaining we can observe that the buffer utilization is initially high and then reaches the peak and then falls down gradually. The peak represents the peak of the chaining process. After the completion of the chaining and the after the video has been fully watched by the client the buffer utilization falls; this is the dip in the graph.

## 5. Conclusion:

In this paper we have proposed a protocol for optimal streaming using the chaining process and clients' residual bandwidth. Through simulation we have proved the advantages of using this protocol in the real world environment. We have shown the behaviour of the system

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

and its response to the various real world situations. The protocol uses the clients' uplink bandwidth, the server bandwidth is optimised and also the clients uplink bandwidth that was previously under used is well utilised.

We intend to build on the current system and make some improvements to make the rejection ratio even more lesser as future enhancements. This will include procedures that will consider the popularity and the cost of streaming, in order to dynamically maintain the videos in the proxy and the servers.

## 6. Implications and Applications:

The implications of the proposed protocol can be categorised into three main categories based on the following parameters:

- Bandwidth:
  Our protocol provides a method to conserve the server bandwidth and optimize its use, by utilising the clients' residual bandwidth.
- Cost:

  The cost of streaming is reduced drastically since the more expensive resources like the server and proxy bandwidth are conserved.

- Efficiency:

  The efficiency of streaming is increased since the rejection ratio is lesser when compared to conventional streaming. This is because fewer requests are rejected due to lack of bandwidth, since most of the requests are served by the chaining process.

## 7. Limitations:

The algorithms used in this protocol are not equipped to handle fault tolerance. In case the connection between any two clients, that are involved in a chain, breaks, our algorithm does not provide a mechanism to overcome the drawback.

## 8. References:

[*Wiki-1*] Wikipedia, http://en.wikipedia.org/wiki/Audio_Video_Interleave, http://en.wikipedia.org/wiki/RealMedia, http://en.wikipedia.org/wiki/QuickTime, http://en.wikipedia.org/wiki/MPEG-4, http://en.wikipedia.org/wiki/MPEG-2, http://www.crutchfieldadvisor.com/S-IqrQwjxSpri/learningcenter/home/fileformats.html, Choosing a Suitable Digital Video Format, www.ukoln.ac.uk/qa-focus/documents/briefings/briefing-25/briefing-25-A5.doc

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)

[*Chain-2*] Te-Chou Su, Shih-Yu Huang, Chen-Lung Chan, and Jia-Shung Wang, “Optimal Chaning Scheme for Video-on-Demand Applications on Collaborative Networks”, IEEE Transactions on Multimedia, Vol. 7, No. 5, October 2005

[*Prefix-3*] Bing Wang, Subhabrata Sen, Micah Adler, and Don Towsley, “Optimal Proxy Cache Allocation for Efficient Streaming Media Distribution”, IEEE Transactions on Multimedia, Vol. 6, No. 2, April 2004

[*Optim-4*] Gerassimos Barlas and Bharadwaj Veeravalli, “Optimized Distributed Delivery of Continuous-Media Documents over Unreliable Communication Links”, IEEE Transactions on Parallel and Distributed Systems, Vol. 16, No. 10, October 2005

[*Hybrid-5*] Mohamed Hafeeda and Bharat Bhargava, “A Hybrid Architecture for Cost Effective On-Demand Media Streaming”, FTDCS 2003

[*CBC-6*] Bernard Jules and Mohamed Hefeeda, “pCDN: Peer-Assisted Ccontent Distribution Network”, CBC Technology Review, Issue 4 – July 2007

[*WB-7*] Warner Bros. on BitTorrent

[*Cisco-8*] Cisco IP/TV 3400 Series Servers User Guide

[*HPPro-9*] HP ProLiant BL480c Server Blade

[*Wiki-10*] Wikipedia - IPTV Overview

[*Wiki-11*] Wikipedia - Video Formats

[*Mael-12*] Maelstrom VoD Servers

[*Birds-13*] Birds-Eye.Net, Video on Demand(VoD)

[*Charl-14*] Charlie Davies, John Delaney, “IPTV & VoD market analysis”, July 2005

[Merg-*14*] Video Formats & Bandwidth, www.merging.com

This paper is licensed under Creative Commons Attribution Non-Commercial License (CC-BY-NC)