HSM: A Hybrid Streaming Mechanism for Delay-tolerant Multimedia Applications

Annanda Th. Rath[1]), Saraswathi Krithivasan [2]), Sridhar Iyer [3])

Abstract

*Traditionally, Content Delivery Networks (CDNs) deploy proxy servers at strategic locations at the edge of the network to efficiently serve client requests. With the tremendous growth in multimedia applications and the number of clients accessing such applications, an edge proxy server itself may serve clients connected to it through a multi-hop network of heterogeneous links. Further, a special class of multimedia applications that can tolerate start up delays is emerging. In such applications, client***s** ***require*** *a minimum acceptable* ***quality*** *(loss-free transmission at a minimum encoded rate r_i) and the start of play back at a specific time ($t + d_i$) where t is the current time and d_i is the* ***delay tolerance*** *acceptable to client i. Our work deals with enhancing performance of such networks through a* ***Hybrid Streaming Mechanism*** *(HSM). In HSM a client's request triggers the selection of an intermediate node as a* ***streaming point*** *to which multimedia contents are* ***dynamically*** *transferred from the proxy/source, and this streaming point streams the contents to the client. Transferred contents are temporarily cached at the streaming point to service future requests for the same content. HSM helps a Content Service Provider's objective of satisfying as many client requests as possible and providing enhanced quality to clients given their delay tolerance. Simulation results demonstrate that by leveraging the delay tolerance of clients, and by combining the dynamic download and streaming mechanisms, HSM performs better than directly streaming from edge servers, serving on an average 40% more client requests.*

1. Introduction

For large scale multimedia data dissemination, Content Delivery Networks (CDNs) are used to overcome the limitation of streaming server capacity and link bandwidth constraints in the network. The main objectives of CDNs are to: (i) minimize the start up latency (the time it takes for a client to start the play back) (ii) reduce network congestion, and (iii) reduce the load on the central server [4][12]. CDNs achieve these objectives through strategic deployment of proxy servers, where contents are cached in anticipation of future requests. Each proxy server itself serves as a source for clients connected to it through a multi-hop network of heterogeneous links. Also, streaming media applications are emerging where multiple clients access the contents at specific times according to their convenience. In these special class of multimedia applications, termed *delay-tolerant* applications [5], clients request for the multimedia content specifying their *requirements*, (i) stream *quality* -- a minimum rate at which they want to

[1] Ex-Masters' student at Kanwal Rekhi School of Information Technology, IIT Bombay, India. Currently at: Department of Computer Science and Communication Engineering, Institute of Technology of Cambodia, Cambodia, annanda@it.iitb.ac.in
[2] Kanwal Rekhi School of Information Technology, IIT Bombay, India, saras@it.iitb.ac..in
[3] Kanwal Rekhi School of Information Technology, IIT Bombay, India, sri@it.iitb.ac..in

receive the stream, and (ii) *delay tolerance* -- the time they will wait for the play out of the stream.

Universities offering their courses to a set of global subscribers, multinational corporations providing training to employees across cities are some examples. Note that mechanisms proposed in the literature to efficiently serve requests for multimedia content, including CDNs, propose ways to *minimize* the start up delay [1][2][3][6], whereas we deal with applications that may *require* start up delay. In this paper, we present a *Hybrid Streaming Mechanism* (HSM) to increase the efficiency of Content Service Providers (CSPs) by using a combination of dynamic download and streaming mechanisms. In HSM, a client's request triggers the selection of an intermediate node to which multimedia contents are *dynamically* transferred from the source, and this *streaming point* streams the contents to the client. Transferred contents are temporarily cached at the streaming point to service future requests for the same content until the contents need to be evicted.

Simulation results of HSM show that by leveraging the delay tolerance of clients, and by combining the dynamic download and streaming mechanisms intelligently, HSM performs better than direct streaming from edge servers, serving on an average 40% more client requests. In the next section, we present a motivating example. We present the HSM algorithm in Section 3 and present our experimental analysis in Section 4. Section 5 presents the conclusions of our work.

2. Motivating example

We consider a network modeled as a tree, with source S at the *root* and the clients C_1, C_2,..,C_{14} at the *leaves*, as shown in Figure 1.

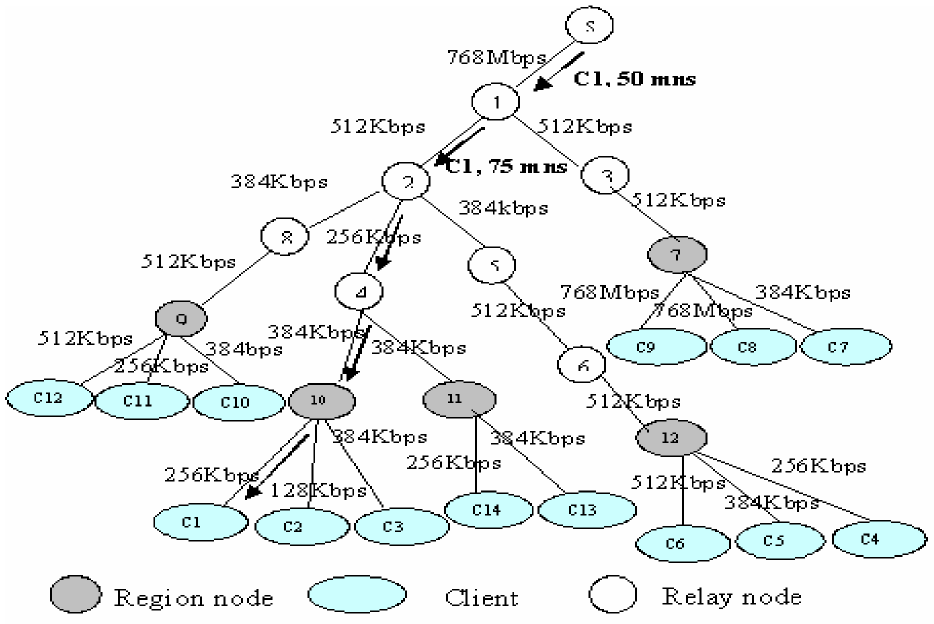

Figure 1: Sample tree network topology

All other intermediate nodes serve as *relay nodes*. A relay node that directly serves a group of clients is termed a *region node*. We use the term *region* to refer to the sub tree that contains the region node and the clients it serves. For example in Figure 1, the network has 5 regions with nodes 7, 9, 10, 11, and 12 serving as region nodes. We refer to the network from S to the region nodes as the *backbone* of the content dissemination network. While most existing research focus on the Internet (best effort network) as the underlying network [10], we assume that the backbone is *provisioned* (using Multi-protocol Label Switching (MPLS)) to support multimedia content dissemination. Consider a stream of play out duration 2 hrs. Client arrival times and requirements are shown in *Table 1*.

Table 1: Details of clients requesting the stream

Clients	Request arrival time (λ) (Mins.)	Client requirements	
		Minimum rate (α) (Kbps)	Delay tolerance (δ) (Mins.)
C1	0	256	30
C14	+10	256	60
C6	+75	256	30
C9	+75	480	15
C12	+75	256	30

Let us consider the case when S is streaming:

- C1 arrives at time zero. It gets 320 Kbps (equation used is derived in Section 3.2), which flows along the path (S-1-2-4-10-C1).
- This path is occupied for 2 hrs at 320 Kbps. Some links are underutilized in this case. The underutilized links and their unused bandwidths are given below: (i) Link (S –1): 448 Kbps; (ii) Link (1-2): 192 Kbps, and (iii) Link (4-10): 64 Kbps.
- C14 joins the network at time t=10. Since C14 shares links (S-1-2-4) with C1, its request cannot be serviced.
- Client C6 joins network at t=75. It shares links (S-1-2) with C1. Given its delay tolerance, C6 can get only a stream rate of 240 Kbps. Since this rate is below C6's minimum required rate, request from C6 is also rejected.
- Similarly, clients C9 and C12 also get rejected. *Thus, when the source streams directly, only one out of five clients is serviced by the CSP.*

Suppose HSM is used. When a request arrives at the central server, it determines the stream rate that can be provided to the client given the client's delay tolerance requirement and the location of the streaming server, termed *streaming point*. The central server then starts downloading the data to the chosen streaming point and allows it to stream the contents to the clients. The data sent by source to the streaming point is also cached at that node for a short duration, in the interest of future requests for the same content. A detailed discussion of HSM algorithm is presented in Section 3. In the rest of the paper, we use the term *Pure Streaming Mechanism* (PSM) to refer to direct streaming from the source.

- As before, the deliverable stream rate at C1 is 320 Kbps. But now we choose node 4 as the streaming point.(Details of streaming point selection are presented in Section 3.3)

- Data is transferred from the source to the streaming point along the path (S-1-2-4). Note that all links except link (4-10) are fully utilized in this case.
- C14 joins the network 10 minutes after C1. Since C14 shares links (S-1-2-4) with C1, it is not possible for C14 to immediately initiate a new stream from S. However, since C14 is requesting for the same streaming object, as the object is being cached at node 4, its request can be serviced from node 4. C14 gets the stream at 320 Kbps which is greater than its minimum rate requirement.
- Clients C6, C9, and C12 join the network at time t=75. Before t=75, C1's transmission across links S-1 and 1-2 are finished and these links become free. All three clients C6, C9, and C12 get serviced with a stream rate of 480Kbps, their streaming points being at nodes 5, 1, and 8 respectively. *As a result, under HSM all 5 clients can be serviced.*

Thus, we observe that HSM performs better than PSM in terms of number of serviced clients. This is because, in HSM, links from the source to the streaming point are freed sooner than PSM, as the link bandwidth is fully utilized. Another important feature of HSM is that future requests for the same content from other clients in the sub tree can be serviced from the cache at the streaming point. We use a simple caching mechanism requiring limited memory and a simple cache eviction policy with very little overhead. This property allows HSM to further improve the number of serviced clients.

3. Details of HSM

In this section, we first present the HSM algorithm in Section 3.1.We present details of the algorithm including (i) Streaming point selection and (ii) Expressions used in HSM, in Sections 3.2 and 3.3 respectively.

3.1. HSM Algorithm

When request from a client is received, source S invokes the HSM algorithm. The main idea of the algorithm is as follows: find the best rate at which the client can be serviced given its delay tolerance and the link bandwidths in the path from the source to the client. A feasibility check is made to ensure that the client can be serviced with its minimum rate requirement. Then, if the links are free, find the streaming point, transfer the contents to the streaming point and start streaming from the streaming point. Time to Live of the Content is initialized. Otherwise check to see if any node in the client's path (selected as streaming point for a previously admitted client) has the content in its cache. If cached content is available, the Time to Live of the Content (TTLC) is updated, and content is streamed to the client, else the client's request is rejected. The algorithm is outlined in *Figure 2*.

3.2. Expressions used in HSM

In this section, we present the equations for the time to transfer the file from the source to the streaming point and the deliverable rates at clients. We use *Figure 3* to derive the expressions.

Figure 2: HSM Algorithm (invoked by the Source S)

```
When a client's request arrives,
 /* client specifies minimum rate required ($\alpha$ ) and its delay tolerance($\delta$) */
Given, $\alpha$ and $\delta$, and the streaming duration SD and the weakest link $L_{min}$ in the client's path, determine
the deliverable stream rate at the client : $SR= L_{min} +( L_{min} * \delta /SD)$ /* Equation (1) from Section 3.2.1
is used */
If $SR < \alpha$
          Reject request
Else
          If link is free
                /*Streaming point (SP) selection*/
                    Find the bandwidth of weakest link in the client's path from source to region node,
                    $B_{min}$
                    If $SR <= B_{min}$
                              Choose SP as the relay node with maximum number of outgoing links
                    Else
                              Choose SP as the node below $B_{min}$
                    End
                    Find time to transfer the contents to SP, $T_t$=filesize/$B_{min}$
                                                        /* Equation (2) from Section 3.2.2 is used */
                    Update the client's delay tolerance and calculate the deliverable stream rate.
                    Transfer contents from S to the selected SP and start streaming
                    Intialize Time to Live of the Content (TTLC) /* Refer to Section 3.2.3 */
          Else
                    If the requested content is already cached at SP
                              Update Time to Live of the Content (TTLC) /* Refer to Section 3.2.3 */
                              Accept request and stream from cache
                    Else
                              Reject request
                    End
          End
End
```

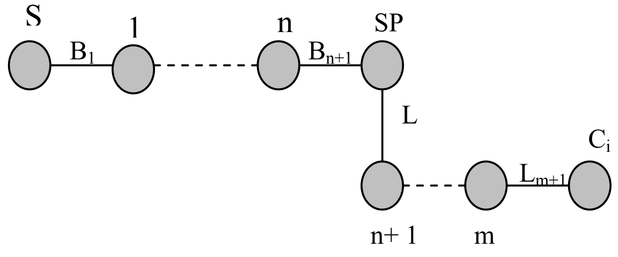

Figure 3: Example used to illustrate the derivations

S	Source
SP	Streaming Point
C_i	Client i
1,…,n, n+1,…,m	Relay nodes
B_i	Link bandwidth of link i from S to SP.
L_i	Link bandwidth of link i from SP to C_i.

3.2.1. Equation for deliverable stream rate at a client

With reference to *Figure 3*, let L_{min} be the minimum of link bandwidths $L_1, L_2,\ldots, L_{m+1}$, in the path between SP and client C_i. Let d_i be the delay tolerance of C_i and let SD be the total duration of the stream. The deliverable stream rate at client C_i is given by the expression:

$$\mathbf{SR_i= L_{min} +(L_{min} *d_i/SD)} \qquad (1)$$

We derive the expression as follows:

- When the stream is encoded at L_{min}, C_i receives it without any loss.
- However Ci waits for a time d_i before the play out starts. During this waiting time, an amount of data can be streamed to C_i given by: $L_{min} * d_i$.
- The amount of extra data that C_i gets per second is $L_{min} * d_i/SD$. Thus, the delivered stream rate at C_i, is $SR_i = L_{min} + (L_{min} * d_i/SD)$.

3.2.2. Time to transfer to streaming point

As shown in *Figure 3*, let there be n relay nodes 1, 2,…, n from source S to the streaming point SP. Let $B_1, B_2, \ldots, B_{n+1}$ be the link bandwidths in the path from S to SP. Time to transfer the file from S to SP is dictated by the weakest link in the path between S and SP, B_{min} and is given by:

$$\mathbf{T_t = filesize/B_{min}} \qquad (2)$$

3.2.3.Time To Live of Content (TTLC) in the cache

We use a simple method to determine the value of Time To Live of Content (TTLC) such that the cache management has very little overhead unlike the replication strategies used by CDNs in their core network [8].
Consider C_i with delay tolerance d_i requesting for a stream with duration SD.

- The client's transmission starts at time = t_0+d_i.
- The client finishes its transmission at (t_0+d_i+SD).
- Hence the stream needs to be active for the duration d_i+SD.

We choose this value as the TTLC for the stream in the cache at SP. When multiple clients access the same stream at the same time, we choose the maximum of the delay tolerance values of the clients in the above expression.
When there is a new request for the same stream before the TTLC expires, it is extended to $T_c + d_k - (T_c - t_k) + SD$, where T_c is the TTLC of the current content, t_k is the time when C_k's request arrives and d_k is the delay tolerance of C_k.

3.3. Streaming Point Selection

In HSM, a selected relay node serves as the steaming point for all the clients in its sub tree instead of the central server. Several methods have been proposed for caching proxy locations in the context of a CDN [11]. Here, we select the streaming point based on the following criteria: (i) streaming point should help to improve the number of serviced clients and /or (ii) the position of the streaming point should help to improve the stream rate for other requests which come from the region serviced by that streaming point.
Let SR_i be the deliverable stream rate at client C_i having requirements: minimum rate of α Kbps and delay tolerance of δ minutes. Let R_i be the region node serving C_i. Let B_{min} be the bandwidth of the weakest link in the path from S to R_i.
Case 1: When SR_i is less than or equal to B_{min}, we select the relay node in the client's path from S to R_i with the most number of out going links as the streaming point. Rationale for this strategy is as follows:

- In this case, the stream will flow without introducing any delay up to R_i. Hence, any node in the client's path can be chosen as the streaming point.

- However, when the relay node with most out going links is chosen, more clients can be serviced concurrently.

Case 2: When SR_i is greater than B_{min}, one of the nodes below B_{min} in the client's path from S to R_i is chosen as the streaming point. Rationale for this strategy is as follows:

- Weak link in a client's path uses up the client's delay tolerance.
- When one of the nodes below B_{min}, is chosen as the streaming point, other clients' requests in the sub tree made within TTLC may be serviced with better stream rates, as the stream's flow is not subjected to this weak link.
- As in case 1, while selecting a node below B_{min}, the node with most out going links is chosen.

3.3.1. An illustration of streaming point selection

We consider the same simple network model given in *Figure 1*.
With reference to *Table 1*, consider the request from C_1 arriving at time zero.

- C1 allows a delay tolerance of 30 minutes.
- The deliverable stream rate SR_1 for C1 is 320 Kbps.
- This rate is greater than B_{min1}, 256 Kbps in the path from source to the region node 10 serving C1. Hence we choose the streaming point at node 4.

As explained in Section 2, all five clients are serviced when node 4 is chosen as the streaming point.

Table 2: Details of clients requesting the stream

Clients	Request arrival time (λ) (Mins.)	Client requirements	
		Minimum rate (α) (Kbps)	Delay tolerance (δ) (Mins.)
C2	0	128	90
C4	+15	128	30
C11	+15	128	30
C14	+15	128	60

Table 2 provides another instance of client arrivals and their requirements. Consider the request from C2 arriving at time zero.

- C2 specifies a delay tolerance value of 90 minutes.
- Stream rate SR_2 that can be delivered to this client is 224 Kbps.
- This rate is less than B_{min2} (256 Kbps) in the path from source to the region node 10 serving C2. Hence we choose the streaming point at node 2.

When node 2 is chosen as the streaming point, requests from clients C4, C11, and C14 arriving 15 minutes later can be serviced concurrently from the cache at node 2, even though the links (S-1) and (1-2) are occupied by the stream serving C2.

4. Performance Evaluation

In this section, we compare HSM with the *Pure Streaming Mechanism* (PSM), the term we use to refer to direct streaming from the source, under various network topologies and client requirements using Matlab. The following performance metrics are used: (i) the number of serviced clients and (ii) percentage improvement of client's stream rate as compared with its minimum rate requirement.

4.1. Simulation parameters

The following parameters remain the same across all our experiments:
(i) Multimedia play out duration is set to 2 hours, (ii) Without loss of generality, queuing delay and propagation delay are set to zero, and (iii) Period over which client arrivals are monitored, termed *observation period*, is set to 4 hours. The key factors that affect the performance of streaming mechanisms are: (i) network topology with specific link bandwidths, (ii) client request arrivals, and (iii) clients' requirements. To understand the impact of these factors, we consider 100 different topologies. For these topologies, we first keep the client requirements constant and vary their arrival rates; then we keep the arrival rate constant and vary their delay tolerance requirements.

- The first 50 topologies termed as *Class 1* have high link bandwidths from source to region node. The bandwidths are chosen randomly from the range (256 – 768 Kbps).
- The next 50 topologies termed as *Class 2* have low bandwidth (weak links) in the backbone, from source to region node. The bandwidths are chosen randomly from the range (128 – 256 Kbps).
- All topologies have total number of nodes in the range 100 to 500, where the number of nodes is selected randomly.

4.2 Details of experiments

We study the impact of the key factors on the two metrics - the number of serviced clients and percentage improvement of client's stream rate - under PSM and HSM below:

4.2.1. Uniform client delay tolerance values, varying client arrivals:

Clients' delay tolerance values are set to 30 minutes. Arrival rate of the clients' requests is varied from 1 to 30 per minute. Clients' minimum rates are set to 128 Kbps. We observe both the parameters – number of serviced clients and stream rate improvement at the clients - in *Figure 4*. In this figure, X-axis represents the number of client requests per minute and Y-axis (on the left) represents the percentage of clients serviced and secondary Y-axis (on the right) represents percentage of stream rate improvement. When Class 1 topologies are used; *Figure 4* shows that when the request rate increases, the number of serviced clients decreases for both the mechanisms. This is as expected. However, the decrease is more pronounced in PSM compared to HSM. For example, when the request rate is 10 per minute, HSM services around 80% of the requests while PSM services only around 50%. Note that the difference between the number of serviced clients in PSM and HSM keeps widening as the number of requests increases. These results show that HSM is attractive for a CSP.While comparing the percentage improvement in stream rates with reference to *Figure 4*, we observe that PSM appears to provide clients with better stream rates compared with HSM. This is because percentage improvement in stream rates is calculated only for the serviced clients. Since PSM rejects 78% of client requests (compared with only 30% for HSM), its stream rate improvement seems better still only marginally.

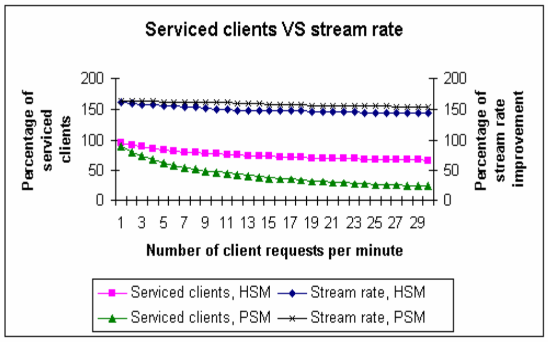

Figure 4: Percentage of serviced clients VS. Percentage of stream rate improvement (Class1)

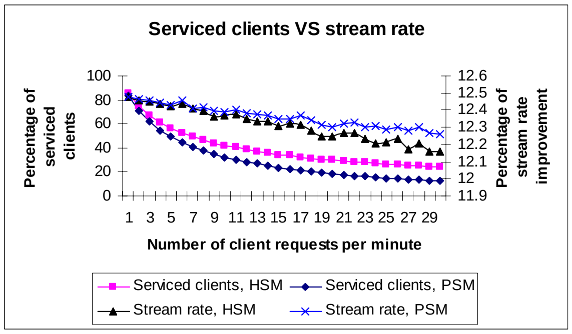

Figure 5: Percentage of serviced clients VS. Percentage of stream rate improvement (Class2)

Figure 5 presents similar results for class 2 topologies. We observe that HSM performs marginally better than PSM. In class 2 topologies links from the source to the region nodes have low bandwidths. In this case, transferring the file to a relay node does not provide any advantage, as the time for transferring the file is the same even when streaming server is placed at the source. The main advantage of HSM is that by choosing a streaming point appropriately, future requests from clients for the same content can be serviced from the cached contents, which contributes to the slightly better performance of HSM.

4.2.2. Uniform arrival rate, varying client delay tolerance values:

Client arrival rate is kept constant at one request per second. Delay tolerance values are set to 30, 60, 90, and 120 minutes respectively for the experiments. Clients' minimum rates are set to 128 Kbps.

Due to space limitation, we only present results for Class 1 topologies, to evaluate the impact of clients' delay tolerance on the number of serviced clients using PSM and HSM. Results in *Figure 6* demonstrate that as clients' delay tolerance increases, the performance of HSM gets better. When the client delay tolerance is equal to the streaming duration, HSM services nearly 100% of the clients' requests. In the case of PSM, as shown in *Figure 7*, client delay tolerance has very little effect on the number of client requests serviced. This is an interesting observation that HSM is especially beneficial for delay-tolerant multimedia applications where the CSP's backbone has provisioned links, as the available bandwidth is better utilized in this mechanism.

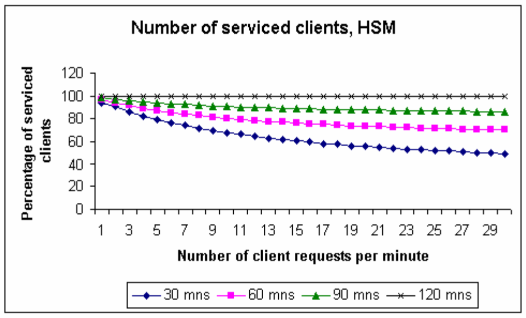

Figure 6: Impact of client delay tolerance values on HSM (Class1)

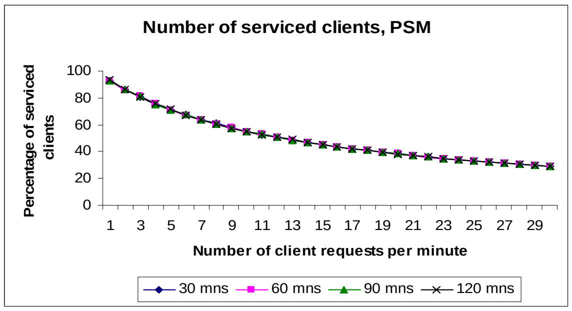

Figure 7: Impact of client delay tolerance values on PSM (Class1)

5. Conclusion

Typically in a content dissemination network controlled by a CSP, weak links are at the edge of the network closer to the clients. By using a combination of dynamic download and streaming mechanisms, provisioned links in the CSP's backbone can be fully utilized, serving more client requests when compared to a centralized server handling all the streaming requests. HSM, the proposed hybrid streaming mechanism, uses this idea to improve the performance and hence the revenue for a CSP, leveraging the delay tolerance specified by the clients. We have shown that by choosing appropriate relay nodes as streaming points, on the average 40% more requests can be serviced using HSM as compared with PSM. Our on-going research includes efficient utilization of combination of resources [7][9] such as streaming servers, buffers, and transcoders to maximize revenues for a CSP in delay tolerant multimedia applications.

References

[1] P. Frossard, O. Verscheure, Batched patch caching for streaming media. IEEE communication letter, vol.6, no. 4, pp.159-161, April 2002.

[2] K.A. Hua, Y. Cai, S. Sheu, Multicast technique for true video-on-demand services, ACM Multimedia, pp. 191-- 200, 1998.

[3] J. Liu, X. Chu, J. Xu, Proxy cache management for grained scalable video streaming. In proceedings of IEEE INFOCOM, 2004.

[4] G. Pallis, A. Vakali, Insight and perspectives for content delivery networks, Communications of the ACM, vol. 49, no.1, pp. 101-106, 2006.

[5] S.Krithivasan, S. Iyer, Startegies for efficient streaming in delay-tolerant multimedia applications, IEEE ISM 2006, to be held at San diego, U.S.A., December 11-13, 2006.

[6] S. Sen, J. Rexford, D. Towsley, Proxy prefix caching for multimedia streams, In Proceedings of the Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, 1999.

[7] B.Shen, S.-J. Lee, S. Basu, Caching strategies in transcoding-enabled proxy systems for streaming media distribution networks. IEEE Transaction on Multimedia, vol. 6, no. 2, pp. 375-386, June 2000.

[8] S. Sivasubramanian, M. Szymaniak, G. Pierre, M. Steen, Replication for Web hosting systems, ACM Computing Surveys, vol.36, no.3, pp. 291-334, 2004.

[9] G.-M. Su, M. Wu, Efficient bandwidth resource allocation for low-delay multi-user video streaming. IEEE Transaction for circuits and systems for video technology, vol. 15, no. 9, pp. 1124--1137, September 2005.

[10] D. Wu, Y. T. Hou, W. Zhu,,Y.-Q. Zhang, J. M. Peha, Streaming approach over Internet: Approaches and directions. IEEE Transaction on circuits and systems for video technology, vol. 11, no. 3, pp. 282-300, March 2001.

[11] J. Xu, B. Li, D. L. Lee, Placement problem for transparent data replication proxy services, IEEE Journal on Selected Areas in Communications, vol. 20, no. 7, September 2002.

[12] Survey of Content Delivery Networks (CDNs) http://cgi.di.uoa.gr/~grad0377/cdnsurvey.pdf