ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 19, April 2017

PROTOCOL ON- DEMAND -VIDEO ON DEMAND

1**Arul Ganapathy M** arulganapathy@outlook.com,
2 **Jaya Kumar S** jayakumar.s@rmp.srmuniv.ac.in
1 PG Scholar, CSE Department, SRM University, Chennai
2 Asst. Professor in CSE Department, SRM University, Chennai

***Abstract*—Videos account for over half of all traffics to mobile phones, internet-connected devices, and this fraction is increasing. As such, improvements in mobile video delivery can potentially lead to significant societal savings. Multicasting is a well-known technique for reducing the load on a network when that load is induced by multiple users (or clients) accessing the same content. Multicasting provides clear benefits when streaming videos of popular live events such as concerts or sports matches. Yet some video contents are viewed "on demand"—when a user joins, he/she wishes to watch the video from the beginning. Existing approaches are not suitable for the less number of users and providing all video segments from the disk which are slow. We propose a dynamic protocol switching based on the number of users actively accessing the server and we predict the next segment of the video accessing by the user and provide it from the main memory instead of fetching it from the disk which make the processing faster and other packets send in the compressed format. While playing the available segment, in the meantime, other compressed segments get downloaded and decompressed in the client. Servers will handle the priority signal with a separate thread.**

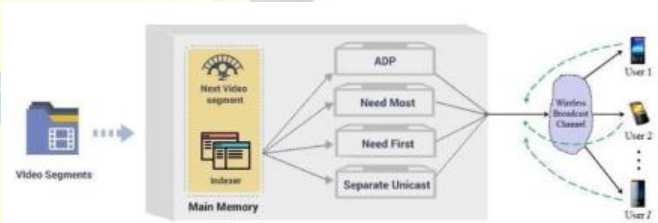
I. INTRODUCTION

Streaming videos over the multicast group in an effective way is explained over in this paper. Scheduling the video as segments using approximation dynamic programming is explained in the previous paper. Approximation programming is suitable for the large number of the users. Packet loss may occur for small number of users. So, we are going to switch the algorithm in a dynamic based way on the number of active users accessing the video segments. We are going to maintain the collection list Indexer to maintain the active users and list of users accessing the same video segments. Indexer will maintain in the main memory using the LRU. Next significant segments which are going to accessed by the users are identified by the indexer and make it available in the memory of the server. If more number of video segments are needed by the user which are more than the size of main memory, most needed video segments will get loaded. For scheduling the video segments, we can use the various algorithm, each algorithm has different advantages. We are going to utilize those at the time they are needed. Yes, we are going to switch the algorithm which is going to schedule the video segments based on the needs.

II. EXISTING SCHEMES

Approximation Dynamic programming was used to schedule the video segment to the user. Instead of sending all the feedbacks to the server, feedbacks of this system were given to the server in partial to avoid the unnecessary processing. This approach supports the larger number of the users and packet loss may be possible on small number of the users.

III. ARCHITECTURE



IV. INDEXERS

This is the collection object which is going to maintain the active number of users using the video segments. This segment will available in the main memory with singleton class and store in the disk as parallel. In the case of any loss of data in the memory, again while accessing the indexer it will loaded into the memory.

Eg:

```
List<Index> Indexer = [{"videosegmentid":"id1",
"users":["user1","user2"]},{
"videosegementid":"id7",

"users":["user1"]}]

Public List<Indexer> getInstance(){
        If(indexer!= null)

        return indexer;
        else{

        indexer = getIndexerFromDisk();
        return indexer;
```

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 19, April 2017

} getIndexFromDisk(){//DAL}

V. EXTENSION

In this system, we are mainly concentrating on the system that can provide services based on the need. When load is higher, server will use the high-level scheduler algorithm to schedule the segments. For urgent need, it will provide in the high priority manner using a separate thread. For less number of users, it will use the unicast or push or web socket to provide the segments. For mostly needed segments, server will predict the next segments and load those or reference in to the in-memory for faster access.

A. Initial Stall Time

To reduce the waiting time for the user to load the video, we are using Indexer which is a collection object maintaining the number of user accessing the video segment. If the video is accessed by the more number of the user, Approximation Scheduling Algorithm was used to send the segments to those users. In case of less users accessing the video, separate connection was established for those users to serve better.

B. Faster Response from the server

Using that Indexer, we can have identified the next needed segment and make those videos readily available in-memory.

C. Reduce size of the file transferred over the network

The segments other than in the memory were grouped and compressed were transmitted to the client. From that file size is reduced in the transmission.

D. Priority Response

If clients have no more video segments to play other then the current playing segment, then clients raise the urgent signal to the server and server spawn the new worker thread to respond for the priority signal. So, another separate CPU is available to

taken care of the urgent need

VI. ALGORITHM

As discussed, we are going to switch the algorithm based on the need and the load handle we are going to use the following algorithm to schedule video segments on a different scenario. We also going to use the multithread for responding the urgent need of the video segments.

computing its solution, one simply looks up the previously computed solution, thereby saving computation time at the

expense of a (hopefully) modest expenditure in storage space. (Each of the sub-problem solutions is indexed in some way, typically based on the values of its input parameters, so as to facilitate its lookup.) The technique of storing solutions to sub-problems instead of re-computing them is called "memorization".

Dynamic programming algorithms are often used for optimization. A dynamic programming algorithm will examine the previously solved sub problems and will combine their solutions to give the best solution for the given problem. In comparison, a greedy algorithm treats the solution as some sequence of steps and picks the locally optimal choice at each step. Using a greedy algorithm does not guarantee an optimal solution, because picking locally optimal choices may result in a bad global solution, but it is often faster to calculate. Fortunately, some greedy algorithms (such as Kruskal's or Prim's for minimum spanning trees) are proven to lead to the optimal solution.

Approximation Dynamic Programming is used to divide the complex problem into the sub-smaller problems and try to find the solution for each sub-problem, merge result of those and try to found the solution for the complex one. Likewise, instead to try to concentrate on all the users at the time ADP break down, among the number of the users, take few and find the shortest path to schedule the segment and recursively run the algorithm to find solution for all other nodes.

We use ADP for a larger number of the users. ADP will work fine for the large number of the users and packet loss may occur for the less number of the user.

Let Rserver bits per second be the wireless transmission for the downlink. Since we have defined that the ratio of the transmission rate and the rate of layer i to be

ratio $\triangleq$ server

A. Approximation Dynamic Scheduling(ADP)

Dynamic programming (also known as Dynamic optimization) is a method for solving a complex problem by breaking it down into a collection of simpler sub-problems, solving each of those sub-problems just once, and storing their solutions – ideally, using a memory-based data structure.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 19, April 2017

The channel state changes per the Markovian transition rule. The user receives the video segments transmitted by the transmitter if the channel state is r during time lot of j +1. The location increments by one if the user already has at least the base layer of the next video segment or receives it during time slot j +1, in which case the video stalls and We assume that information about user arrivals and departures is immediately available to the server. In addition, we shall initially assume that all users feedback their complete states at the end of every time slot. We shall lift this assumption later in Section 6 and study the general case in which the feedback is rate limited. We assume that the transmit delay of the feedback is negligible compared to the duration of a time slot. We also assume that the feedback channels are noiseless. The server uses the feedback to schedule which segments (and which layers) to transmit in the next time slot. A sequence of such scheduling decisions constitutes a scheme or a scheduler. Let J be the number of slots in the time horizon. Let the power set of a set A, i.e., the set of all subsets of A, be denoted as P(A). The size of a set A is denoted by A. Similar notation
is used to denote the length of a vector. Mathematically, a scheduling scheme is then given by a mapping

$$: U \rightarrow S$$

where

$$\triangleq \{\{-1,0,1, \dots , -1\} \times \{0,1\} \times \{\ ,\ \} \times \{0,1\}^{\times 2}\} \times \{0,1,2, \dots , -1\}$$

is the set of all possible scheduling the decisions. In the definition of S, A1 and A2 are the sets of indices of base and enhancement layers, respectively, of video segments that are co-scheduled. Note that _ may depend on time. To evaluate the performance of a scheduling scheme, we consider four relevant video performance metrics: the startup delay, the stall time, the number of interruptions (i.e., the number of stalls), and the bit-rate of the video (which could be replaced to some extent by general video quality metrics). [3] discussed about a method, This scheme investigates a traffic-light-based intelligent routing strategy for the satellite network, which can adjust the pre-calculated route according to the real-time congestion status of the satellite constellation. In a satellite, a traffic light is deployed at each direction to indicate the congestion situation, and is set to a relevant color, by considering both the queue occupancy rate at a direction and the total queue occupancy rate of the next hop.

R_{server}.Later we will allow the server to multicast to a subset of clients at a potentially higher rate, in which case patching could also be viewed as a choice of .

B. *Existing Schemes and Simple Heuristics*

We shall compare the performance of schedulers that we develop against one oblivious scheme and several omniscient schemes. Among oblivious schemes we considerPyramid broadcasting [8], which is prototypical.

Among omniscient schemes we consider separate unicast, the needed-first scheduler, and a second simple scheduler that we call needed-most.

C. *Separate Unicast*

Separate unicast exploits the feedback of the client, but it does not use the broadcast nature of the wireless channel. In this scheme, the server transmits the video to a user based on state only, and the amount of video transmitted in a time slot depends on the server's portion transmission rate which is allocated by the user. For more than one user, however, it is suboptimal because it does not exploit multicast opportunities. Note that virtually all the video-on-demand systems having separate unicast, from this we can show our illustration in that how much gain can offer while practice.

In this unicast, separate connection is established with each client connected to that server. Connection context is maintained in the server. Segments were pushed from the server entire state of the client is feedbacked to the server. We use this approach when less number of the user connected to the network and actively using the connection.

We are going to use a Separate Unicast to supports the less number of users and to make way of faster for few connections. But, it will not be consistent when a high range of the users have accessed the video segments.

D. *Need Most*

In this need most segment scheduler, the segment can be arranged based on the client requirement. Thus, a segment that is needed by two clients would be sent before one that is needed by only one. We shall see that this scheduler performs better than needed-first when the number of users is large, which is expected since the needed-most scheduler prioritizes efficient use of the downlink channel. For small numbers of clients, however, the scheduler does not perform well.

While considering the need most high frequency of users accessing the same segments means, we suppose to use the need most scheduling algorithm to serve better and to give a positive approach to the users on the link. Using the help of the indexer we can find out that how many users are accessing the same segments. If most number of the users accessing the same segments particularly, probably that they are going to access that segment for the next part of the link. Since our indexer will fetch the segment from the disk and keep the storage reference in the in-memory for faster accessing by the users.

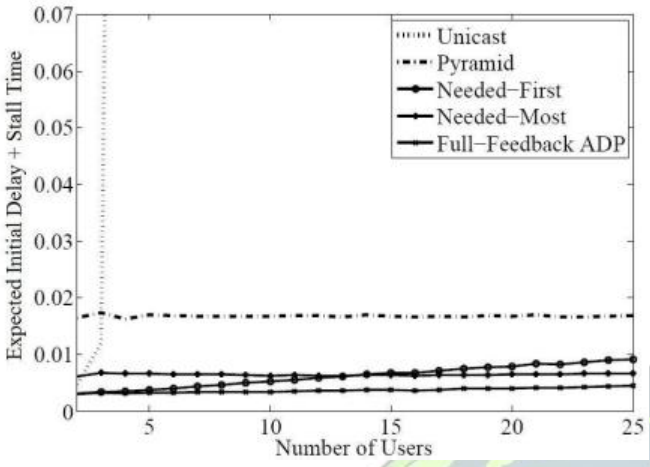
E. *Pyramid*

When the packet transfer rate of the server and client is higher, we can use the pyramid approach where all the video segments were transmitted in each connection. If users need the

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 19, April 2017

segment, the users can access the connection and get the segments. This approach is used when the client has high data transfer rate or high speed internet.



VII. CONCLUSION

In this paper, we have concentrated on four different areas in which we ought to provide the video segments faster to the users accessing the server. We are maintaining the user access list in the in-memory for predicting the next needed segment and make it available in the in-memory and schedule the video segment based on the active user in the network and we switch the algorithm based on the need. We maintain a separate CPU to handle priority requests.

REFERENCES

[1] Cisco Systems, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2014-2019," May 2015. [Online]Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-nextgeneration-network/white paper c11-481360.pdf

[2] P. R. Kumar and P. Varaiya, Stochastic Systems: Estimation, Identification, and Adaptive control. Prentice-Hall Information and System Sciences Series, Englewood Cliffs, NJ: Prentice Hall, 1986.

[3] Christo Ananth , P.Ebenezer Benjamin, S.Abishek, "Traffic Light Based Intelligent Routing Strategy for Satellite Network", International Journal of Advanced Research in Biology, Ecology, Science and Technology (IJARBEST), Volume 1,Special Issue 2 - November 2015, pp.24-27

[4] G. S. Fishman, Monte Carlo: Concepts, Algorithms, and Applications. New York: Springer, 1996.

[5] T.-L. Lin, S. Kanumuri, Y. Zhi, D. Poole, P. C. Cosman, and A. R. Reibman, "A versatile model for packet loss visibility and its application to packet prioritization," IEEE Trans. Image Proc., vol. 19, no. 3, pp. 772-735, Mar. 2010.

[6] A. R. Reibman and D. Poole, "Predicting packet-loss visibility using scene characteristics," in Proc. IEEE Packet Video, 2007, pp. 308-317.

[7] A. Chan, K. Zeng, P. Mohapatra, S. Lee, and S. Banerjee, "Metrics for evaluating video streaming quality in lossy IEEE 802.11 wireless networks," in Proc. IEEE INFOCOM, 2010, pp. 1613-1621.

[8] Md. Saifur Rahman and Aaron B. Wagner "A Downlink Scheduler for Multicasting Wireless Video-on-Demand"