Mesh Based Peer to Peer Live Video Streaming using ANT Algorithm

A. Duraisamy, M.Sathiyamoorthy

Abstract- Data distribution and data retrieval has always been an integral part of the Internet. In particular, video streaming has been gaining popularity over the last few decades. Though there has been no shortage of innovations and Experimentations, no single system has been able to deliver highly scalable and reduce congestion to the requesting users. The development of p2p technologies brings unprecedented new momentum to Internet video streaming. This work proves that p2p is indeed more efficient by taking into account the factors of scalability, response time and reliability of serving the request. This work is broadly and logically divided into joining the network group, requesting, retrieving and video playing segments. The Requester takes care of receiving the file and storing it in the buffer. The video receiver takes care of sensing the user connections and playing the appropriate video on the requesting peer side. To provide video in efficient manner, this work use ACO (Ant Colony Optimization) algorithm and bandwidth for choosing optimized peer. From that optimized peer, user can get video. We also take care of scheduling and scalability issues.

Index Terms — Ant Colony Optimization, Peer to Peer, Mesh network, Video Streaming, Bandwidth Optimization.

I. INTRODUCTION

In recent years high bandwidth broadband connections have become available to everyone, the demands for online video are constantly increasing. Video-on-demand systems like BitTorrent [2] and YouTube [7] are highly successful, where the user has the possibility to In contrast to video on demand; live video streams are only available at one particular time. The most important task here is to create a continuous video stream. The momentary used systems are mainly constructed as a server to client solution, where one single source distributes the video content to its receivers. This approach scales up to certain point; until the server's full capacity is reached. A Unicast system, where each client has a direct connection to the server, has the advantage that it is easy to implement, and live and on-demand streaming is not a problem. The disadvantage the server has to sustain the entire video distribution work load.

This means the server needs a high bandwidth connection and the operator has to carry all the costs of maintaining the functionality of the system. Peer-to-Peer networks now offer the possibility to distribute video content to an unlimited number of users and reducing the bandwidth bottleneck on the source.

Manuscript published on 28 February 2013.

* Correspondence Author (s)
A.Duraisamy, Department of Information Technology, University College of Engineering, Tindivanam (T.N), India.
M.Sathiyamoorthy, Department of Information Technology, University College of Engineering, Tindivanam (T.N), India.

The use of centralized server architecture is not needed since the video distribution throughout the network is handled by each client. Bandwidth savings at the server side can be achieved through P2P video content delivery systems. P2P systems reduce the server's work enormously and make the system more scalable because a new peer joining the network has no longer an impact on the server's resources. A direct server-client solution forces the server to have sufficient bandwidth for each client to stream the video, unlike the P2P approach where the server (source of the video stream) needs only enough bandwidth to stream the video to a certain initial amount of peers. The rest of the video distribution is then handled by the peers in the network.

Today, the traditional client-server approach requires a tremendous amount of effort and resources to keep up with the continuous growth of the Internet in terms of users and bandwidth. The Peer-to-Peer communication paradigm provides a solution for the growing requirements. A definition given in describes a P2P system as follows: "*A peer-to-peer system is a self-organizing system of equal, autonomous entities (peers) which aims for the shared usage of distributed resources in a networked environment avoiding central services.*" P2P networks should be able to satisfy the vast demand for resources such as bandwidth, storage capacity, or processing power and scale accordingly. The distributed nature of P2P system increases reliability in case of failures by replicating data over multiple peers. Over 50% of Internet traffic is due to Peer-to-Peer applications, sometimes even more than 75%.

Video-over-IP applications have recently attracted a large number of users on the Internet. Traditional client server based video streaming solutions incur expensive bandwidth provision cost on the server. Peer-to-Peer (P2P) networking is a new paradigm to build distributed network applications. Recently, several P2P streaming systems have been deployed to provide live and on-demand video streaming services on the Internet at low server cost. In this work, we provide a survey on the existing P2P solutions for live and on-demand video streaming.

Existing approaches to P2P streaming can be divided into two general classes: (*i*) *tree-based approaches* use push-based content delivery over multiple tree-shaped overlays, and

(*ii*) *mesh-based approaches* use swarming content delivery over a randomly connected mesh. Comparison between these two classes has been conducted. And then we compare and contrast the performance of representative protocols from each class using simulations.

We identify the similarities and differences between these two approaches. Furthermore, we separately examine the behavior of content delivery and overlay construction mechanisms for both approaches in static and dynamic scenarios.

The question to solve in a P2P system for video streaming are: How does a peer choose its partner from which it is receiving the video stream. Does a peer have a single source or are the multiple sources for the stream? What is the structure of the P2P network, a tree based network or a meshed based network? In a P2P system is always the risk that a source peer may suddenly fail or disconnect. For example in tree-based system where every peer has a single source, the disconnect of a single peer could damage the video distribution profoundly. The higher a peer is in a tree hierarchy the more peers depend on its wellbeing and its continuous streaming capacity. If one peer disconnects an entire branch of the tree could lose the video stream. To create a stable video distribution system every peer in the system should have multiple sources to circumvent the single point of failure.

Representative P2P streaming systems, including tree, multi-tree and mesh based systems. Peer-to-Peer (P2P) networking has recently emerged as a new paradigm to build distributed network applications. The basic design philosophy of P2P is to encourage users to act as both clients and servers, namely as peers. In a P2P network, a peer not only downloads data from the network, but also uploads the downloaded data to other users in the network.P2P file sharing applications, such as, have been widely employed to quickly disseminate data files on the Internet. More recently, P2P technology has been employed to provide media streaming services. Several P2P streaming systems have been deployed to provide on-demand or live video streaming services over the Internet are introduced.

In a mesh-based P2Pstreaming system, peers are not confined to a static topology. Instead, the peering relationships are established or terminated based on the content availability and bandwidth availability on peers. A peer dynamically connects to a subset of random peers in the system. Peers periodically exchange information about their data availability. Video content is pulled by a peer from its neighbours who have already obtained the content. Since multiple neighbours are maintained at any given moment, mesh-based video streaming systems are highly robust to peer churns. However, the dynamic peering relationships make the video distribution efficiency unpredictable.

II. RELATED WORK

Guo Y, Magharei N, and Rejaie R, et al [1] presented the A Comparative Study of Live P2P Streaming Approaches. In existing approaches of P2P streaming can be divided into two general classes: (*i*) *tree-based approaches* use push-based content delivery over multiple tree-shaped overlays, and (*ii*) *mesh-based approaches* use swarming content delivery over a randomly connected mesh. Comparison between these two classes has been conducted. Our results indicate that the mesh-based approach consistently exhibits a superior performance over the tree-based approach. It provides better protocol to play the video.

The BitTorrent protocol [2] can be used to reduce the server and network impact of distributing large files. Rather than downloading a file from a single source server, the BitTorrent protocol allows users to join a "swarm" of hosts to download and upload from each other simultaneously. The protocol is an alternative to the older single source, multiple mirror sources technique for distributing data, and can work over networks with lower bandwidth so many small computers, like mobile phones, are able to efficiently distribute files to many recipients.

A user who wants to upload a file first creates a small *torrent* descriptor file that they distribute by conventional means (web, email, etc.). They then make the file itself available through a BitTorrent node acting as a *seed*. Those with the torrent descriptor file can give it to their own BitTorrent nodes which, acting as *peers* or *leechers*, download it by connecting to the seed and/or other peers.

Pieces are typically downloaded non-sequentially and are rearranged into the correct order by the BitTorrent Client, which monitors which pieces it has, can upload to other peers and which it needs. Pieces are of the same size throughout a single download (for example a 10MB file may be transmitted as ten 1MB Pieces or as forty 256kB Pieces). Due to the nature of this approach, the download of any file can be halted at any time and be resumed at a later date, without the loss of previously downloaded information, which in turn makes BitTorrent particularly useful in the transfer of larger files. This also enables the client to seek out readily available pieces and download them immediately, rather than halting the download and waiting for the next (and possibly unavailable) piece in line, which typically reduces the overall length of the download.

Garbacki P, Pouwelse D.E.JA, and Sips HJ et al [3] presented the BitTorrent Streaming System. The first proposed system is BitTorrent System (BiToS) for streaming the video in peer to peer (p2p) video streaming systems. File distribution over BitTorrent comparises of a tracker, hosted on a website that maintains a list of the peers, currently downloading the file along with information on the amount of file data each peer has downloaded and uploaded and the amount of the file data they still need to obtain. The peer then contacts the tracker which provides a list of peers and their information. From that information, the peer requests fragments to download from that list of peer.

Helen J. Wang, Philip A. Chou, and Venkata N.Padmanabhan et al [4] presented the Heterogeneity and Congestion Controlmechanism to identify several challenges peculiar to the P2P setting including robustness concerns arising from peer unreliability and the ambiguity of packet loss as an indicator of congestion. We propose a hybrid parent and child-driven bandwidth adaptation protocol that is designed in conjunction with a framework for robustness and that exploits application-level knowledge.

Liu J, Yum T.S.P, and Zhang B.X et al [5] described the Cool Streaming as the core operations (periodically exchanging data availability information and downloading unavailable data from and uploading available data to appropriate peers) are same as in BitTorrent. It provide general idea on efficient scheduling algorithms, yielding better playback and less overhead, robustness of system can be improved by splitting and streaming.

Sen J.S and Towsley D et al [6] presented the Prefix Caching technique for multimedia streaming whereby a proxy stores the initial frames of popular clips. Upon receiving a request for the stream, the proxy initiates transmission to the client and simultaneously requests the remaining frames from the server. In addition to hiding the delay, throughput and loss effects of a weaker service model between the server and the proxy, this is simple prefix caching technique aids the proxy in performing work ahead smoothing into the client playback buffer.

Viewing YouTube [7] videos on a personal computer requires the Adobe Flash Player plug-in to be installed on the browser. The Adobe Flash Player plug-in is one of the most common pieces of software installed on personal computers and accounts for almost 75% of online video material. YouTube accepts videos uploaded in most container formats, including .AVI, .MKV, .MOV, .MP4, DivX, .FLV, and .ogg and .ogv. These include video formats such as MPEG-4, MPEG, VOB, and .WMV. It also supports 3GP, allowing videos to be uploaded from mobile phones. Videos with progressive scanning or interlaced scanning can be uploaded, but for the best video quality, YouTube prefers interlaced videos to be de-interlaced prior to uploading. All the video formats on YouTube use progressive scanning.

In ALMI [12] (Application Level Multicast Infrastructure) and Overcast [10], are a central node coordinates of the tree management, as in CoopNet. In ALMI, a centralized session controller is one that has to gathers peer-to-peer ping data to perform a minimum spanning tree computation and periodically reorganize the coordination of tree. This procedure is not wellsuited to the CoopNet scenario because the minimum spanning tree computation is not desire to low tree depth and high tree diversity. Also, periodic reorganizations may be too disruptive for large trees. In Overcast, the root node plays a central role in tree management, because the root node manages all the child node or slave nodes in a tree, which act as a master of a tree. However, a key difference compared to CoopNet arises because overcast strives to build deep distribution trees that maximize the bandwidth from the root to any node. So that Overcast is intended for use in the context of a dedicated set of infrastructure nodes that is relatively stable.

Recent work has leveraged the scalable routing are feeds it can be provided by distributed hash tables (DHTs) to build efficient multicast trees (e.g., Bayeux [13], and Scribe [9]). It is unclear how well these perform in the face of a high rate of node churn, especially since the data structures needed for efficient routing are updated very lazily. Furthermore, a fundamental difference compared to CoopNet is that in these systems nodes can be called on to forward traffic even if they are not themselves interested in the data. All of the above pieces of work differ from CoopNet in that they seek to build a single distribution tree, so the tree diversity is not a consideration. The only piece of work (besides our previous workshop paper [11]) to our knowledge that supports the use of multiple distribution trees is SplitStream [8].

SplitStream uses multiple trees to evenly distribute forwarding load across the nodes in a tree. This is accomplished by making a node a leaf in all but one tree. We leverage this insightful observation in our new, deterministic CoopNet tree construction algorithm (randomized algorithm [11]). However, SplitStream and CoopNet differed in a fundamental way. SplitStream is built on top of the distributed Scribe protocol [9]. Therefore, it is assumed that nodes will be available to forward traffic even when they are not interested in it. This facilitates the construction of a diverse set of trees since nodes can be retained at specific positions in the trees long after they have “departed”. In fact, whether a node is called upon to forward traffic depends only on its node ID and the multicast group ID. Therefore, it is possible that a root node may be assigned more than one child or slaves it can handle.

SplitStream [8] presents a procedure to address this issue. However, this may sacrifice interior-nodedisjointness. CoopNet, on the other hand, has to construct such trees *incrementally*, i.e., as nodes arrive and depart one by one. This presents a challenge because nodes that arrive early would tend to be at the higher levels in all trees (unless we resort to large-scale and potentially disruptive reorganizations of the trees to move some of the early members down the tree). Our deterministic tree construction algorithm attempts to construct short and diverse trees with minimal disruption (only sterile nodes, with no descendants, get pushed down the tree). The issue of limiting the out-going bandwidth requirement of a node (which is a key concern in SplitStream) is solved trivially in CoopNet because the number of children of a node is explicitly controlled by the omniscient root. Furthermore, our deterministic tree construction algorithm is able to guarantee the disjointness of the set of interior nodes across the trees.

Traditional video-streaming system has a centralized server, through all video gets streamed for every video request, which overloads the server. So we move to Peer to Peer (p2p) video streaming mechanism in which every peer acts as provider and requester of video. In mesh-based p2p network, any peer can join and leave at any time. Suppose if a user requests a video, provider while sending video, may leave the network.

Due to this problem, requester may not obtain full video. Requester peer try to receive video chunks from the peer, having better bandwidth i.e from optimized peer. But this may be a problem for the newly joined peer regarding selection of optimized peers. The optimized bandwidth of the peer can be founded by using the ANT algorithm for path caching. This work provides a set-up that solves the above problem in efficient manner.

The objective of this work is to provide video streaming from multiple sources and merging at the requesting peer to form a complete requested video over P2P networks by implementing Ant Colony Algorithm [1]. Peer to Peer (P2P) networks [6] are typically used for connecting peers via largely ad hoc manner. Sharing content files containing audio, video is very common using P2P technology. A pure P2P network does not have the notion of clients or servers but each peer simultaneously function as both "clients" and "servers" in the network.

This model of network arrangement differs from the client-server model where communication is usually to and from a central server. A typical example of a file transfer that is not P2P is a file transfer protocol server where the client and server programs are quite distinct, the clients initiate the download/uploads, and the servers react to and satisfy these requests [3].

But the problem here is, requester can't predict who has video. In this work, we use Ant Colony Algorithm for efficiently obtain video.

III. PROPOSED WORK

The purpose of this proposed work is to lay down the system architecture and detailed design for the work entitled "Mesh Based Peer to Peer Live Video Streaming Using Ant Algorithm". It contains the overall description of the work.

The peers wishing to request a video, forms a network by getting registered with tracker. Information like IP address, port number, name of the video and amount of video content is provided to tracker which stores that information in database. Tracker gets the bandwidth of peer with the help of WMI Explorer tool. When a new user/peer requests a video, it contacts the tracker.

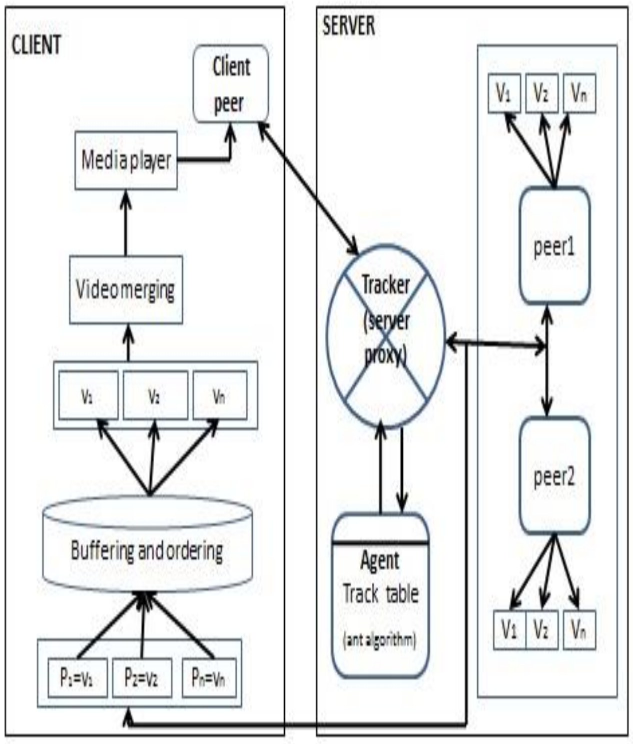

Figure 1: Architecture for video streaming in client-server.

The tracker in turn searches the database for requesting video. If requested video is available in single peer, then tracker sends that peer information to the requesting peer. If multiple peers have that same requested video, then the tracker sorts its peer's information based on bandwidth. Then, the highest bandwidth peer information is transferred to the requester.

Requesters establish a connection to provider and get video through streaming. In some cases, some of the requested video parts are available in one provider peer eg: v1, v2, v3 and remaining parts eg: v4, v5 available in some other provider peer. Then the requester needs to establish a thread to both peer and obtain the video segments in parallel. Video provider information is updated in trace table of peers in the path through which the video is streamed. When a new peer request for video that is already requested, the peer uses the updated information in trace table and gets the video.

The provider splits the video into several parts and assigns a sequence number to each part. The requester while receiving the video chunks; checks sequence number, merge video parts and play it in the media player. If a particular video part is missed, requester waits for it, even if other parts are received. After got the video parts, it merges at the receiver end then played it on using some video player.

IV. IMPLEMENTATION AND DISCUSSION

1. Mesh Formation

Many nodes/peers are connected using Wireless links or LAN cables.Peers in a p2p network could join and leave at any time and rate of peer links vary in time to a great extent. Our system takes care of such transient peers and provides scheduling. Whenever a peer sends a join request, tracker collects all information from the peer and stores it in database and sends keep-on-alive message periodically to all peers in network.

2. Resource Acquisition

2.1 Peer Registration

Registration message send by each peer in mesh network. It means that if any peer wants to join to mesh network then first step is registration phase. Then the peer information can be updated in the tracking system, and then Registered on list maintained by tracker. The trace table helps to identify the peer in the Network, and also it maintain the registered information.

Table 1. Peer Registration with Tracker

IP address	Port no	Resource Contain	Bandwidth	Requested Video
168.254.63.10	640	A.flv	521.606	AR.flv
168.254.63.20	564	AR.flv	524.846	A.flv

Above table describes the information (internet protocol address, port number, resource, bandwidth etc.,) of all active peers in the network. That information is maintained by the tracker. So that the Tracker uses this table in order to provide optimized peer information to the requester.

2.2 Session Initialization

The peers introduce themselves to the other peers in the list provided by tracker by sending HELLO messages to it. Each active peer in the mesh network provides process id and resource information are introduce themselves to other active peers in the mesh network. Then the Session formation of the peers with the other peers who had that video resource is done. An active peer in the chunk has searched themselves to found who has requested video. All the Peers which request to join the session send a P-HELLO message to the other peers. The message format of the P-HELLO message is as follows,

Table 2: Session Initialization

MESSAGE TYPE	P-ID	HOP COUNT
Join	1	2
Request	4	5
Join	3	4

Above table describes the hop-count and peer id's information. Join entry for joining the network and request entry for requesting purpose. An active peer in the chunk has searched themselves to found who has requested video.

2.3 Service Request Processing and Respond

2.3.1 Service Request Process

Requesters establish a connection to provider and get video through streaming. In some cases, some of the requested video parts are available in one provider peer eg v1, v2, v3 and remaining parts eg v4, v5 available in some other provider peer. Then the requester needs to establish a thread to both peer and obtain the video segments in parallel. Video provider information is updated in trace table of peers in the path through which the video is streamed. When a new peer request for video that is already requested, the peer uses the updated information in trace table and gets the video.

The sending peer splits (using mdc splitter) the video into chunks and Assign sequence number to every chunk. Chunks of video reach the requested peer in multiple paths. Pull based mechanism used to obtain only required video and avoid congestion over links.The requester while receiving the video chunks, checks sequence number, and then merge those video parts. If a particular video part is missed, requester waits for it, even if other parts are received. After got the video parts, it merges at the receiver end then plays it on using some video player.

2.4 Retrieving the Video Parts

2.4.1 Video Splitter

The sending peer splits (using mdc splitter) the video into chunks and Assign sequence number to every chunk. The Splitter performs the actual video stream splitting. Since multiple description coding is used, it is needed to split the original stream into several substreams (descriptions). This is the task of the Splitter. The Encoder sends the encoded original video stream and the Splitter creates the independent substreams after a specified MDC scheme. Dependent on the used MDC scheme the two components Encoder and Splitter may be combined into a single component given that the splitting and encoding task can't be done independently. Another task of the splitter it to create the UDP packets to sends the video stream over the network to the target.

2.4.2 Video Merger

This module merges various segments to give the full video as output. The purpose of the Joiner is to take several incoming substreams and combine them to the original stream by using a specific MDC scheme. If the Splitter receives all substreams, the original video can be reconstructed, otherwise only a part of the original stream will be available.

Again, like the Encoder and Splitter component, the Decoder and Joiner may be combined into one component, depending on the used MDC scheme. The Joiner takes the incoming packets from the In-Buffer and, retrieves the video data from the packets and creates the video stream.

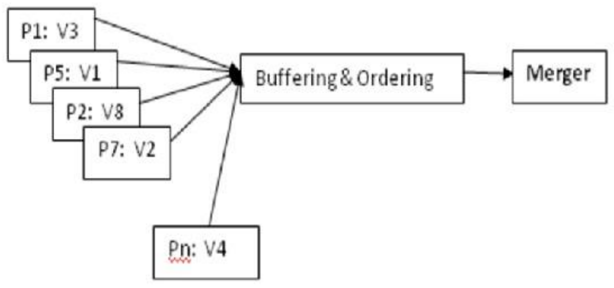

Figure 2. Video Splitter and Video Merger.

2.4.3 Proxy server / Tracker

The tracker Sends keep-on-alive message to know that peers status. That messaged has to help weather the peers are currently active or inactive in the network. Based on the response these peer information updated on the list by tracker. The state of peers keeps tracked and updates the status by tracker. To avoid failure in upload or download a video, due to sudden leaving of peers from that cluster, the tracker sends Keep-on-Alive messages to peer list and records each peer state.

Suppose if a peer not responding within round trip time (rtt), then it calls task schedule module. The Tracker maintains a state table of the peers as follows:

Table 3. State Table of Peers

Process id	Video content	Bandwidth	Requested video	Amount
1	.flv	521.606	AR.flv	full
2	R.flv	524.846	New.flv	full

Above table describes the list of active peers in the network. Suppose, if a provide peer leaves the network suddenly happened then tracker must remove that provider peer information from its table.

2.4.4 Task Scheduling

If any peers leaves the cluster while uploading or downloading the video. Or any failure occurred then the video providers suddenly left from the cluster while video transfer to the video requester. It can simply remove that peer information (peer which leaves) from the list maintained by tracker and re-establish connection to other peer which has same video. If a peer leaves the cluster, then corresponding countermeasure must be taken by tracker node.

Step 1: Remove that peer information (peer which leaves the cluster) from the list and update the lists which are maintained by tracker.

Step 2: For further video request, it redirect to alive peers who had that requested video content.

Repeat the above steps for all the transient peers.

2.4.5 Load Tracing

The peer requesting for video, requests the tracker to connect to peer that contain requested video.

Step 1: Peer requests tracker for video content.

Step 2: Tracker has information about peer which contains video. Tracker sends list of optimized peer's id to requested peer.

Step 3: Requesting peer establish a connection with that optimal peer which containing video and streaming begins.

2.4.6 Choosing Optimized Peer

Optimal peer based on higher bandwidth with the help of WMI EXPLORER tool. Once obtain request from requesting peer, Tracker is responsible for giving information about peer who had requested video and higher bandwidth. Using that information, requested peer obtain video.

2.4.7 Path Caching

Peer requests for same video that is already streamed. This already selected optimal path is used for video streaming of other peers streaming of video. Within some period of time, if another peer requests for same video, the same optimal paths can be used by the requested peer with the help of TRACE table by ant-based algorithm.

2.4.8 ANT Algorithm

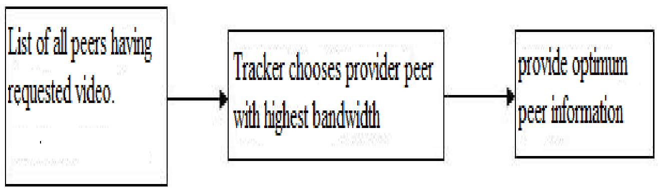
List of all peers having requested video.
Tracker chooses provider peer with highest bandwidth
provide optimum peer information

Figure 3. The Ant Algorithm used in Track table.

System is initiated by requesting peer, which is requesting the video to the tracker, where tracker has a one, which acts as a proxy server, to provide the corresponding response to the requesting peer. Tracker maintaining the all the peers in the network and also maintain the resource named as track table, which maintain the all the active peers in the mesh network, if any peer may leave from the network by failures then the tracker should know who are they alive now in network by sending keep-on-alive message to each peers in the mesh network, if no response from any one peer then it can simply discard that information from the track table. The video is provided to any peer in the network then it can choose the shorted path between them by ant algorithm, use of this algorithm to finds the optimum peer information among the list of all peers having the requested video from the tracker.

V. CONCLUSION

Thus a mechanism has been designed by which a system which tries to reduce load on the server by directly getting the video from the nearby peers in an efficient manner, by caching the previous optimal peer paths in a separate table and throughput captured. A peer can easily search for the video it wants by using ant based search algorithm. This architecture is completely decentralized and hence all peers are of equal importance. Many features can be enhanced and improved in our work. Due to insufficient time some of the modules are not developed. In case we can improve the security issues with highly secured passwords and can provide authentication only to the trusted users. Hence the load can be distributed to every other peer in network so the load and traffic for the video can be reduced gradually. GUI can be further developed with inbuilt video player that can increase outlook of our work. Searches for the video content can improved in efficient manner.

REFERENCES

[1] Guo Y, Margharei N, and Rejaie R, 2007, " Mesh or multiple tree: A comparative study of live p2p streaming approaches", In *Proceedings of IEEE INFOCOM* , pp. 1424 – 1432.
[2] BitTorrent, http://www.bittorrent.com
[3] Garbacki P, Pouwelse D.E.JA, and Sips HJ, February 2005 "The bittorrent p2p file-sharing system: Measurements and analysis," in *4th International Workshop On Peer-To-Peer Systems (Iptps).*
[4] Helen J. Wang, Philip A. Chou, and Venkata N. Padmanabhan Microsoft Research: " *Supporting Heterogeneity and Congestion Control in Peer-to-Peer Multicast Streaming*".
[5] Liu J, Yum T.S.P, and Zhang B.Xd, March 2005 "CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming," in Twenty Fourth Annual joint Conference of IEEE on Computer Communication Societies, pp. 2101-2111.
[6] Sen J.S and Towsley D, March 1999 "*Proxy prefix caching for multimedia streams*," in Eightieth Annual joint Conference of IEEE on Computer Communication Societies, pp. 1310-1319.
[7] YouTube, http://www.youtube.com
[8] Castro M, Druschel P, Kermarrec A-M, A. Nandi, Rowstron A, and Singh A. February 2003 SplitStream: High-bandwidth Content Distribution in a Cooperative Environment. In *Proceedings of IPTPS.*
[9] Castro M, Druschel P, Kermarrec A-M, and Rowstron A. October 2002 SCRIBE: A Large-scale and Decentralized Application-level Multicast Infrastructure. *IEEE JSAC*, 20(8):pp.100–110.
[10] Jannotti J, Gifford D, Johnson K.L, Kaashoek M.F, and O'Toole J.W. October 2000 Overcast: Reliable Multicasting with an Overlay Network. In *Proceedings of OSDI.*
[11] Padmanabhan V.N, Wang H.J, Chou P.A, and Sripanidkulchai K. May 2002 Distributing Streaming Media Content Using Cooperative Networking. In *Proceedings of NOSSDAV.*
[12] Pendarakis D, Shi S, Verma D, and Waldvogel M. ALMI: March 2001 An Application Level Multicast Infrastructure. In *Proceedings of USITS.*
[13] Zhuang S.Q, Zhao B.Y, Joseph A.D, Katz R.H, and Kubiatowicz J.D. April 2001 An Architecture for Scalable and Fault-tolerant Wide-Area Data Dissemination. In *Proceeding of NOSSDAV.*

Mr.A.Duraisamy received his M.E degree in Computer Science and Engineering from College of Engineering Guindy, Anna University Main Campus Chennai, India in 2010 and B.E degree in Computer Science and Engineering from Anna University, Chennai, India in 2006. He is currently working as Assistant Professor in University College of Engineering, Tindivanam, Tamilnadu, India. His areas of interest are: Web Application Security, Image Processing, Networking, Bio-Metrics and Data Base Management System. He is published two papers in international journals, one international conference, two national conferences and Life Member in Indian Society for Technical Education.

Mr.M.Sathiyamoorthy received his M.E degree in Computer Science and Engineering from College of Engineering Guindy, Anna University Main Campus Chennai, India in 2007 and B.E degree in Computer Science and Engineering from Madras University, Chennai, India in 2003. He is currently working as Assistant Professor in University College of Engineering, Tindivanam, Tamilnadu, India. His areas of interest are: Web Services, Cryptography and Network Security, peer to peer Networking and Data Base Management System. He has worked in Tata Consultancy Services Ltd, Chennai as Assistant Systems Engineer for 2.5 years.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)
© Copyright: All rights reserved.