Hindawi Publishing Corporation
Advances in Multimedia
Volume 2016, Article ID 5429187, 14 pages
http://dx.doi.org/10.1155/2016/5429187

Research Article

Enhancement of Video Streaming in Distributed Hybrid Architecture

Soumen Kanrar[1,2] and Niranjan Kumar Mandal[2]

[1] *Vehere Interactive Pvt. Ltd., Calcutta, West Bengal 700053, India*
[2] *Vidyasagar University, West Bengal 721102, India*

Correspondence should be addressed to Soumen Kanrar; soumen.kanrar@veheretech.com

Received 25 November 2015; Revised 20 January 2016; Accepted 20 January 2016

Academic Editor: Martin Reisslein

Pure Peer to Peer (P2P) network requires enhancing transportation of chunk video objects to the proxy server in the mesh network. The rapid growth of video on demand user brings congestion at the proxy server and on the overall network. The situation needs efficient content delivery procedure, to the video on demand viewer from the distributed storage. In general scenario, if the proxy server does not possess the required video stream or the chunk of that said video, then the same can be smoothly and rapidly streamed to the viewer. This paper has shown that multitier mesh shaped hybrid architecture composed of P2P and mesh architecture increase the number of requests served by the dynamic environment in comparison with the static environment. Optimized storage finding path search reduces the unnecessary query forward and hence increases the size of content delivery to the desired location.

1. Introduction

The efficient traffic flow of information is a key element in current technology and business environment. This traffic flow is controlled by a complex network architecture and supported communication infrastructure. High-speed network transport mechanism serves as enabling technologies for new classes of communication service such as multimedia and video on demand. The smooth transport of chunk video objects over the network is a challenging issue. Lots of literature address this issue with various concepts based on content driven network (CDN), pure Peer to Peer network, and hybrid as mix of "CDN and Peer to Peer network" [1]. How to design an efficient P2P VOD system with high utilization of bandwidth and low maintenance cost remains an open challenge [2]. The demand for streaming video over a low bit rate channel is increasing in a fast pace for many applications like the newscast, video conferencing, distance learning, video game, entertainment, and so forth. Some sort of traffic and congestion control, admission control, packet drop, needs to be considered due to the characteristic of the video traffic. However, a major bottleneck is the timely delivery of a large amount of data through a very limited bandwidth. The issue directly involves how efficiently the overlay network adapts to the dynamic topology. The video content is pushed from the origin storage server to the peer nodes in the tree-shaped overlay network. Conceptually tree based overlay network is the address followed by multitree streaming approach tree [3, 4]. Here the topology dynamically changes with the multiple subtrees in place of one single tree [4, 5]. However, the mesh based chunk delivery approach gives the better result for both video on demand and the live video streaming. The latter concepts give better performance as observed in PPlive and Livesky [6]. In mesh shaped overlay network, the peers pull "video objects" from the neighboring peer [7]. The pull operation is done by the buffer map between the peers. To achieve the quality of service, chunk of "video object" delivery delays have to be handled efficiently. Quality of Experience (QoE) is defined in [8] as "the overall acceptability of an application as perceived subjectively by end user." In general, HTTP streaming has a bad QoS (quality of service) but a good QoE (Quality of Experience), because HTTP used reliable

mode of transportation. This so happened for QoS (quality of service). The considered parameters are packet loss rate and packet delay and so forth. So it is highly required to enhance HTTP/TCP streaming as a major technology for media transformation [9]. In adaptive bit streaming, the media file is fragmented into small segments or chunks of same duration with the request to some specific time unit [10]. During the adaptive bit streaming, each chunk is decoded independently enabling seamless transport from one quality to another when network conditions are being changed frequently. The playout of a chunk is finished. Video player can start playing the next chunk with a different quality. So, for the adaptive bit streaming, lossy compression is a better option for video on demand [11]. The delay occurs due to the search for a chunk of "video object." On the other side, delay occurs due to the propagation delay and transmission delay from the neighboring peer. So the quality of service mainly depends on fast and correct path search. To find the desired video stream by using the mechanism of minimum hop counts towards the storage peer and to avoid the unnecessary query forwarding as well as to minimize the inter server gossiping during the chunk transfer [12, 13], the major routing protocols for wireless ad hoc networks have traditionally focused on finding paths with minimum hop count (MHC). However, such paths can include slow or lossy links, leading to poor throughput [14]. The energy efficient routing algorithm in wireless networks typically selects minimum cost multihop paths. But those energy aware routing algorithms select a path with large number of small distance hops in varied power transmission environment [15]. The quality of service depends on the aggregate stationary bit rate which exceeds the minimum threshold level from the viewer perspective [16]. This paper is structured as follows. Section 1 introduces the problem. The basic, video encryption mechanisms have been discussed in this section, followed with video objects formation. Section 2 presents the chunk of video object transfer over the hybrid architecture. Sections 3 and 4 present the required architecture and simulation environment of the problem with conclusion at the end.

(1) Video Traffic Burst in Network. The repeatedly occurring theme relating to traffic transportation in broad-band network and high-speed network is the traffic "burstiness." It is exhibited by the key services such as compressed video and file transfer. Burstiness occurs in the course of traffic flow, due to the presence of several relatively short interarrival time sequences. The peak rate (R_{peak}) is an important requirement parameter used during the connection setup time. The other major traffic descriptors are the mean burst period "b," the average bit rate "m," and utilization factor "γ" which is defined as $\gamma = m/R_{\text{peak}}$. The peak rate ($R_{\text{peak}}$) yields the mean bit rate m and the variance σ^2 of the bit rate. The mean of the burst period "b" is used to describe how bursts occur from sender side. It is used to discriminate between different connection setups for streaming from different sources. During multiplexing, it has the same peak and mean bit rate but displays a different behavior. The multiplexer can be modeled as a single queue of length x and N servers each with fixed streaming rate c [17, 18]. The buffer overflow probability for finite buffer size is ε. It can be obtained for given x, c, and N. So the connection setup equivalent bandwidth W for the multiplexed connection is the smallest value of c, for which the buffer overflow probability is smaller than ε. The approximate upper bond of the equivalent bandwidth w_i (for single isolated connection) depends upon the parameters (x, R_{peak}, γ, b, and α), where α is the source capacity of streaming. The equivalent bandwidth of the multiplexed connection can be presented as $W = \sum_{i=1}^{N} w_i$. As it is considering the approximated upper bound for w_i, the equivalent capacity is definitely overestimated. To eliminate the constraint, let us consider a stationary estimation derived from a bufferless fluid flow model. The fluid traffic model gives out with individual traffic units. Instead, it views traffic as a steam of fluid, characterized by a flow rate such as bits per second. The traffic count is replaced by a traffic volume. The bufferless fluid flow model for the stationary approximation exhibits the equivalent bandwidth W. W is selected to ensure that the aggregate stationary bit rate B exceeds W only with the probability smaller than ε. In general expression, it can be presented as $\rho(B \succ W) \leq \varepsilon$, where B can be determined from the stationary distribution of the number of active sources. B can be modeled by the continuous time Markov chain as it potentially captures traffic burstiness, due to the presence of nonzero autocorrelation in the interarrival time sequence [19].

(2) Video Encryption Mechanisms. Video is the collection of still images. The collected set of still images is well orderly sequenced. The compressing technique is used to compress each individual still image of the collected set of images. Encoding technique is used to encode the image independently according to the sequence in the collected set of images. The Joint Photographic Experts Group (JPEG) format is used to compress the still images of the set of images. The sequences of still images are arranged in increasing order independently and individually. The MPEG-4 is the popular coding scheme that encodes the complete frames of video. The MPEG-4 is a collection of coding tools and maintains simple profiles. The most current implementation for temporal scalability is using H.263, MPEG-4, and secular temporal subband coding [20]. The MJPEG format is the motion JPEG video compressed and encoded using JPEG. Each video signal that carries information contains a definite amount of redundancy. So the video sequence contains redundancy. The aim of the video compression is to remove the redundancy from the video signal. Generally, two types of compression technique exist. One type is the lossless compression technique and the other type of technique is lossy compression technique. The lossless compression technique removes the statistical redundancy that is present in the video signal. When the transported video signal at the received end is reconstructed, it presents an identical copy of the original video. The current available technology supports compressing the video signal up to some modest level only. This is one of the greatest handicap legacies for the lossless compression technique, particularly for the storage and transported video signal over the network. This consumes extra space and bandwidth of the underlay network [21–23]. In lossy compression,

we can compress a huge number of video signals, but the reconstructed video signal at the received side is not identical to the original. The lossy compression meets a given bit rate for the storage and can maintain adaptive bit rate during the transmission of the video signal over the network or the internet. Different CODECs run into the storage as well as at the client sites. So, for adaptive bit streaming, lossy compression is a better option for on demand video streaming. The aim of video source coding is bit rate reduction for storage and transmission. The compression is done by removing the different types of redundancy, spatial, temporal, or frequent. Each image is usually divided into many blocks, each of size 8 pixel × 8 pixels. These 64 pixels are then transformed into frequency-domain representation by using what is called discrete cosine transform (DCT) [24, 25]. The frequency-domain transformation clearly separates out the low-frequency components from high-frequency components. Conceptually, the low-frequency components capture visually important components, whereas the high-frequency components capture visually fewer striking in components. The goal is to represent the low-frequency (or visually more important) coefficients with higher precision or with more bits and the high-frequency (or visually less important) coefficients with lower resolution or with fewer bits. Since the high-frequency coefficients are encoded with fewer bits, some information is lost during compression, and hence this is referred to as "lossy" compression. When inverse DCT (IDCT) is performed on the coefficients to reconstruct the image, it is not exactly the same as the original image but the difference between them is not perceptible to the human eye.

The current enhancements in 2D (two-dimensional) video coding are classified mainly into two categories, nonscalable and scalable, respectively [26]. The nonscalable coding, for example, is H.264/AVC (advanced video coding) and H.265/HEVC (high efficiency video coding). The scalable video coding, for example, is H.264/AVC with SVC (scalable video coding) extension. In nonscalable coding bit rate is not reduced, but in scalable coding bit rate is reduced. Generally, H.264/AVC is using an intracoding concept about the prediction of a block inside a frame. Here the neighboring block within the same frame is considered. H.264/AVC provides more flexibility with the blocks of 4 × 4 samples and blocks of 8 × 8 samples for transformation. H.264/AVC is using fundamentally advanced intercoding concept for predicting *B*-frames from the set of highly eligible candidates of past and future *B*-frames. H.264/AVC has 10 modes. The modes are eight angular modes, one DC mode, and one planer mode, respectively. The angular prediction interpolates from reference pixels at locations based on the angle. The DC constant value is an average of neighboring pixels (reference samples). Planer is the average of horizontal and vertical prediction. The new key features of H.264/AVC are enhanced motion compression, small block of transform coding, improved deblocking filter, and enhanced coding. H.265/HEVC is the successor to H.264. H.265 is meant to double the compression rates of H.264, allowing for the propagation of 4 K and 8 K content over existing delivery systems. H.265/HEVC used the concepts of frame partitioning "luminance pixels" into coding tree blocks of sample sizes 16 × 16, 32 × 32, and 64 × 64, respectively. It is flexible and smoothly partitioned into multiple variables sized coding blocks. H.265/HEVC is using updated intracoding procedure for merging of small partition of coding trees. A wide range of intraframe predication is used. H.265/HEVC is massively improving the flexibility stricture of transformation which matches the code tree block structure that is 4 × 4 up to 32 × 32. H.265/HEVC improved the error resilience by using video parameter set (VPS). This concept is used for signaling essential syntax information for the decoding. H.265/HEVC has 35 modes. The modes are 33 angular modes, one DC mode, and one planer mode, respectively [26, 27].

(3) Video Object Formation. The DCT coefficients of each 8 × 8 pixel block are encoded with more bits for highly perceptible low-frequency components and fewer bits for less perceptible high-frequency components. This is achieved in two steps. First step is quantization, which eliminates perceptibly less significant information, and the second step is encoding, which minimizes the number of bits needed for representing the quantized DCT coefficients. Quantization is a technique by which real numbers are mapped into integers within a range, where the integer presents a level or quantum. The mapping is done by rounding a real number up to the nearest higher integer, so some information is lost during this process. At the end of quantization, each 8 × 8 block will be represented by a set of integers, many of which are zeroes because the high-frequency coefficients are usually small, and they end up being mapped to 0. The goal of encoding is to represent the coefficients using as few bits as possible. This is accomplished in two steps. Run length coding (RLC) is used to do the first level of compression. Variable length coding (VLC) does the next level of compression [12, 20]. After quantization, the majority of high-frequency DCT coefficients become zeroes. Run length coding takes advantage of this by coding the high-frequency DCT coefficients before coding the low-frequency DCT coefficients, so that the consecutive number of zeroes is maximized. This is accomplished by scanning the 8 × 8 matrices in a diagonal zigzag manner. Run length coding encodes consecutive identical coefficients by using two numbers. The first number is the "run" (the number that occurs consecutively), and the second number is "length" (the number of consecutive occurrences). Thus, if there is N consecutive zeroes, instead of coding each zero separately, RLC will represent the string of N zeroes as $[0, N]$. After RLC, there will be a sequence of numbers and VLC encodes these numbers using a minimum number of bits [20]. The technique is to use the minimum number of bits for the most commonly occurring number and use more bits for less common numbers. Since a variable number of bits are used for coding, it is referred to as variable length coding [21, 22]. In video encoding, the order in which frames are coded is not the same as the order in which they are played. Thus, it is not necessarily true that the reference frame for the future frame is the one just before it in the playing sequence. In fact, the video encoder jumps ahead from the currently displayed frame and encodes a future frame and then jumps back to encode the next frame in display order. Sometimes, the video encoder uses two "reference" frames, the currently

displayed frame and a future encoded frame to encode the next frame in the display sequence. In this case, the future encoded frame is referred to as P-frame ("predicted" frame), and the frame encoded using two reference frames is referred to as B-frame ("bidirectionally" predictive frame). However, there is a problem for this approach because an error in one of the frames would propagate through the encoding process for ever. To avoid this problem, the video encoders typically encode one video frame (periodically) using still-image techniques and such frames are referred to as "intraframes" or I-frames [28]. A sequence of frames from one I-frame to another I-frame is referred to as a group of pictures (GOP). A GOP is best represented using a two-dimensional matrix. The order of encoding and the order of display are different for the frames in a GOP. It is transported over an IP network just like any other type of data transmission [5]. Typically, videos have three types of frames, I-frames, P-frames, and B-frames, where I-frames capture most of the information in a scene followed by P-frames and B-frames, which capture the "deltas" from the original frame (captured in I-frame) resulting from motion. Finally, I-, P-, and B-frames are packetized in the encoded format as a sequence of frames, which itself is video objects. When delivery is requested, the chunk delivery of "video object" is done from the storage server to the neighboring requested peer nodes. The MPEG-4 looks after the collection of video objects (VO) [24]. The video object is an entry that the end user is able to manipulate. It is an area, which can be of any shape and may exit for an arbitrary length of time. The instance in which the video object occurs is called the "Video Object Plane" (VOP). In the traditional view, we can say that the VOP is a single frame and a set of frames forming a VO. The VOP is in irregular shape and occupies only a part of a frame. The separate objects are coded independently with different resolutions and qualities. These coding schemes adopt a linear uniform sampling scheme disregarding the varying semantic importance of different frames or segments. The characteristics of video traffic depend on the quality of service (QoS) of video traffic.

(4) Arrangement of Peer Nodes. The hierarchy is created by distributing peer nodes to the different levels as illustrated in Figure 1. The levels are numbered sequentially with the highest level of the hierarchy being level zero (denoted by L_0). Each level is partitioned into a set of clusters. Each cluster is composed of peer nodes. The size of each cluster varies within k to $3k - 1$, where k is a constant [29]. The cluster is forming with the set of peer nodes. Those are close to each other. Every cluster has one "cluster head." The cluster head also logically belongs to the lower level; that is, a cluster head of level L_i logically belongs to level L_{i-1}. The cluster head has the minimum distance (with respect to hop count) to all other peer nodes in that cluster at the same level. If the size of a set of peer nodes has an upper bond $(2k - 1)$ then during dynamically splitting for cluster formation, the size of the set reaches up to $2k$. This happened, since at least one higher-level cluster head logically belongs to the lower level. So the set splits into two clusters of size k. The data is transported from one level to another level through the cluster head.

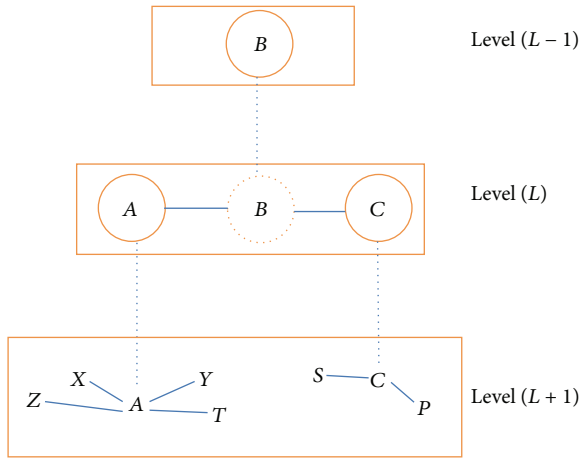

Figure 1: Hierarchal arrangement of peer nodes.

By default, the storage server stayed at the level L_0, which logically belongs to every level. The choice of a cluster leader is very important during the joining of new peer node to the system at the specific level in the hierarchy. The new peer node required minimum number of query messages for finding its position. During the video chunk transportation, formed multicast tree is overlaid to the hybrid architecture. If there is N number of peer nodes in the hierarch, the number of level is bounded with $\Theta(\log_k N)$, where $k \succ 3$ is a constant.

According to Figure 1, A and C are the cluster heads at the level (L). Those cluster heads physically stayed at the level (L), but cluster head is logically part of the level $(L - 1)$. Set of nodes $\{X, Y, Z, T\}$ are directly communicated with the logical cluster head A. Similarly, the set of nodes $\{S, P\}$ are directly communicated with the logical cluster head C for exchanging data and information. This type of hierarchal peer nodes distribution efficiently handles the problem regarding the live streaming of media in large P2P network [30]. The multicast tree construction and efficient clustering of peer nodes based on a hierarchy of bounded cluster size are efficiently improving the performance. The neighbors on the levelwise topology can exchange periodic soft state refreshed without generating high volume of control message. The hierarchal architecture successfully reduces the overburden traffic load to the system. The other important advantages include the peer node departure and smooth cluster head section and cost-effective maintenance of the cluster.

2. Chunk Video Object Transfer over Hybrid Architecture

Figure 2 presents the physical configuration of the overall structure of the video on demand system. Clusters of viewers are symbolically presented by the viewer node. The cache proxy server is connecting viewer nodes to the next level of peer nodes in the multitier architecture. The cache proxy server is installed at the highest level in reverse order in the proposed model of multitier architecture. The proxy cache servers are not directly connected to each other. They are connected to each other via next level peer nodes.

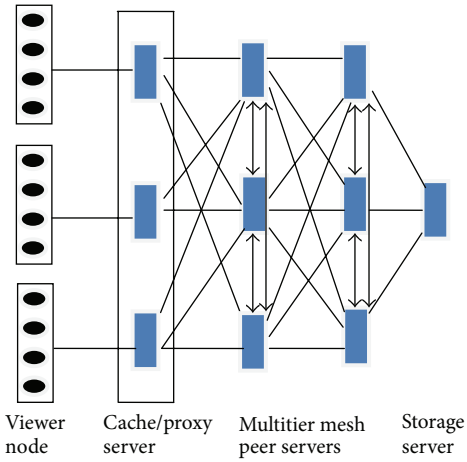

Figure 2: Distributed hybrid architecture.

All the levels including the lowest level, that is, the video storage server, form the multitier mesh shaped architecture. The work of cache proxy server at the highest level is to import the chunk of video objects from next level peer nodes.

Figure 3 presents the working mechanism of chunk transfer. The architecture is the composition of local and global network. The cache proxy server is the part of the local as well as the global network. $P_1, P_2, P_3, \ldots, P_N$ are the peer nodes belonging to different level in the multitier architecture. P_N node position is fixed for the main video storage. The initial request tries to serve locally from the local cache proxy server.

If the local proxy server cannot serve the chunk video object, then the request is broadcast to the neighboring peer nodes of the next level. In this fashion, the chunk request is forwarded among the peers. If any of the peer nodes possess the chunk video objects, that also rebroadcast the message chunk availability, then chunk video object is transferred according to the buffer map procedure. This is the mesh push mechanism from the sender peer node and mesh pull from the receiver peer. Now the copy of chunk video object is stored at the local cache proxy server before delivery to the viewer node. The preliminary survey of the literature shows that more than 10^6 numbers of user's node can concurrently decode the live video streaming within the range of bits rate 400 to 800 Kbps [31]. Current HDMI H.264 video encoders for live streaming and broadcasting use bit rate 250 Kbps to 10 Mbps. Dynamic page replacement at the cache memory of the proxy server can enhance the performance of the video on demand [32] for the "analysis and implementation of large scale video on demand system" [5]. The literatures [33, 34] show that proper cache memory can reduce 50% to 60% of P2P traffic load. Simultaneous viewers are asynchronous; that is, any viewer can access any part of that video at any instance of time. Due to the playback option in video on demand system, a good number of sequential video objects are always available to the media player buffer of the viewers [35–38]. In general, the viewer assesses that video through the cache proxy server. The video objects are a collection of video frames that feed the media player after sequential arranging from the received buffer of the viewer. If the complete video is not available to the cache proxy server, the unavailable part or parts of the chunk block or blocks are imported by the "buffer map via mesh pull approach" according to Figure 3. It uses the multichannel cache proxy server from any numbers of peers at any level in the multitier architecture. The similar buffer map concepts are used in the chunk transfer between the peers in the multitier mesh shaped architecture. The previously proposed video on demand architecture can be classified into some major categories.

(i) Closed loop system: the client retrieves the video segments from the server.

(ii) Prefix caching assisted periodic broadcast: it is a combined effect of open loop and closed loop system.

(iii) Client equipped with set-top box: the set-top box stores locally part of some popular video.

(iv) Peer to Peer network used for video on demand system: the video segments are transported through the network according to peer node requirement.

In this paper, the proposed "distributed hybrid architecture" is a combination of P2P and mesh type network architecture. The cost of video object's transfer through this structure successfully reduces both cases of static and dynamic environment.

3. Analytic Architecture

Let $C^{\dagger}_{(i)}$ be the equivalent capacity of network at session i. The minimum required bandwidth to ensure the aggregate stationary bit rate approaches $C^{\dagger}_{(i)}$ with the considerable frame loss probability η. The aggregate stationary bit rate is required to approximate the adaptive encoding run at the client side video playback. Aggregate bit rate maintains the video resolution within a certain standard. In a particular session, n is the numbers of links that are active and simultaneously transfer the chunk of video objects from the peer nodes through the proxy servers in the multitier architecture. If the transferring stream or "chunk contents" are presented by the random vector X, for $i = 1, \ldots, n$, $E(X_i) = \mu_i$ is the mean transferring rate for individual ith viewers, without the loss of generality. We can assume that X_i are the mutually independent random variables. The aggregate content transfer for a particular session is $B = E(\sum_{i=1}^{n} X_i)$ and $B_n = \mathrm{var}(X_1 + X_2 + \cdots + X_n)$; this includes the smooth video objects transmission at interactive session like playback, moves forward, and so forth. So far, nothing is assumed about the behavior of B_n for indefinite values of n. According to the Chebyshev inequality for the small positive numbers ε and η (previously assumed), the least number of active links is k^* to be found for all n, such that

$$n > k^*. \tag{1}$$

Now,

$$p\left\{\left(\left|\frac{X_1 + \cdots + X_n}{n} - \frac{B}{n}\right| < \varepsilon\right)\right\} \geq 1 - \frac{B_n}{n^2\varepsilon^2}. \tag{2}$$

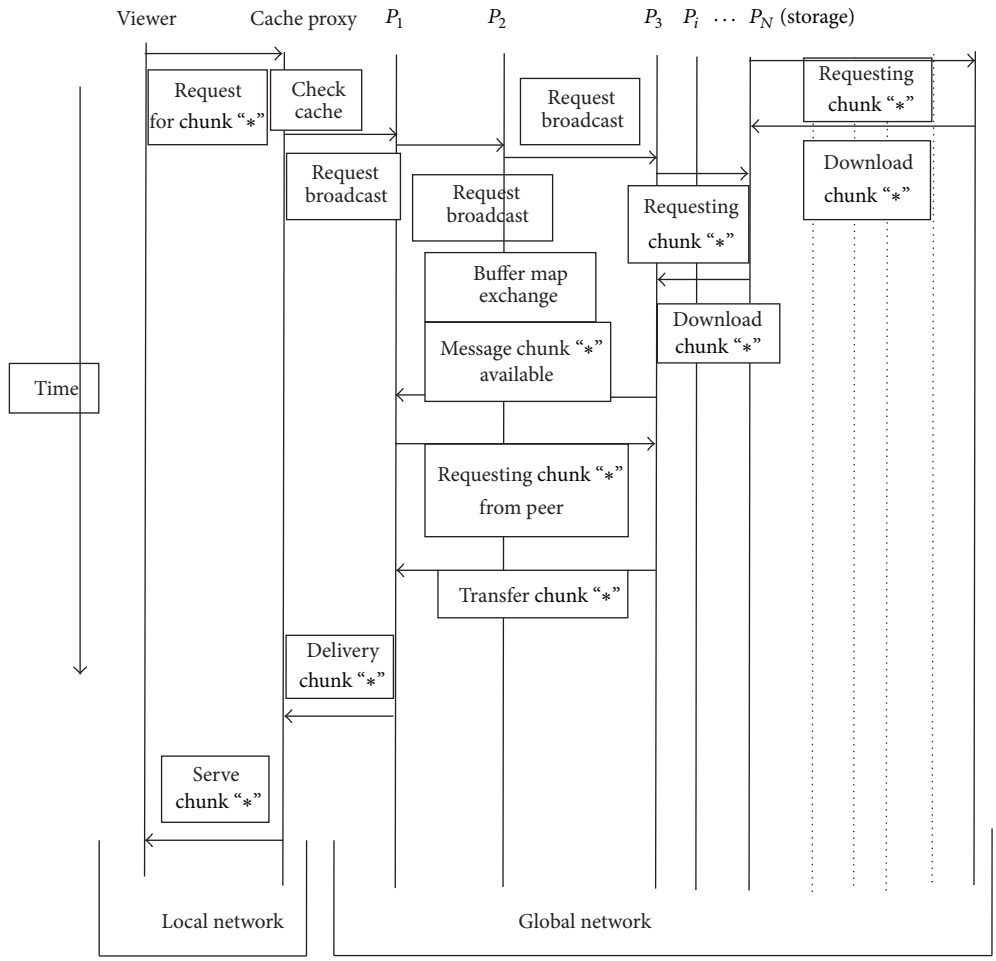

Figure 3: Chunk transfer working mechanism.

So $B_n/n^2\varepsilon^2 \prec \eta$ and $\lim_{n\to\infty}(B_n/n^2\varepsilon^2) \to 0$. Here η is the approximate frame loss probability in a session. All the individual and shared individual links use the maximum bandwidth to transmit the chunk content. Clearly, $\sum_{i=1}^{k^*} \max(E(X_i)) \le C_{(i)}^{\dagger}$. Here $C_{(i)}^{\dagger}$ is capacity of the network. Since η is the packet loss or bit loss probability, it maintains the aggregate stationary bit rate streaming which remains the same for the short burst period. Without the loss of generality, all the above inequality holds true for the event $(B \succ C_{(i)}^{\dagger})$. The probability for the event is

$$\left(B \succ C_{(i)}^{\dagger}\right). \tag{3}$$

So

$$p\left(B \succ C_{(i)}^{\dagger}\right) \le \eta. \tag{4}$$

We can consider it as

$$p\left(B \le C_{(i)}^{\dagger}\right) \le (1-\eta). \tag{5}$$

The distribution of expression (3) can be obtained from the 2-state Markov chain. For every viewer node, there exists at least one active channel, including the case of broadcasting or multicasting of video data objects from the cache proxy server [32, 39]. It contains the full or part of the required video at any level of the pear nodes in multitier architecture. The video content is fully available to the cache proxy server. The proxy server is the part of mesh architecture, which is placed at the maximum level in hierarchy of the multitier architecture at strategic location. The proxy servers are not the peer server, so there are no interlinks between them. The main reason behind it is for the requirement of security and the billing process for commercial purpose also. The billing servers are installed at the level l_1 that is at level one in multitier architecture. Level zero that is l_0 is the position for video storage server. The proxy cache memory is updated or refreshed by the "least frequency frequently used" concepts [32, 40–42]. l_0 is the level for video storage server and height level l_{l+1} is for cache proxy server. If λ is the average fraction of full streaming at a session that each of the supplying peers shares or contributes during that session, clearly $0 \prec \lambda \prec 1$. Let N be the total number of peer nodes including the main storage servers in the multitier architecture. N_c is the upload capacity at level 0 of the mesh type Peer to Peer network. Let $P(l)$ be the sum of the number of peer nodes at the level

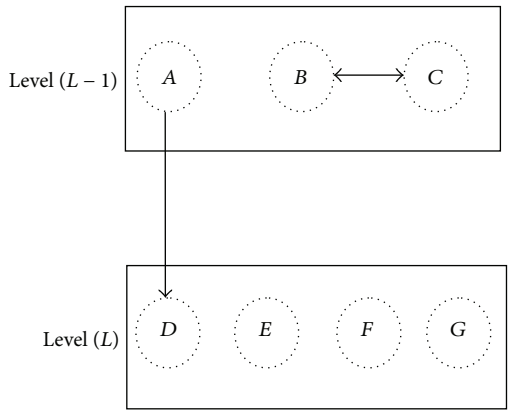

Figure 4: Dynamic cluster connections.

$(l+1)$ and $P(0) = N$. Now $N(l)$ is the upload capacity of the peers at the level l. So in a balanced state of traffic flow during the smooth chunk video objects transfer at any level depends on just lower-level peer's nodes of the multitier architecture. In a particular session of synchronous chunk transfer, the upload capacity of the peer nodes at level l is equivalent to the product of the average fraction of full streaming with the upload capacity of the participating peers at the level $(l-1)$.

This is expressed by $N(l) \approx \lambda N(l-1) \Rightarrow N(1) = \lambda N(0) = \lambda N_c$. In real scenario, peer nodes at the higher level (l) get chunk video objects when the lower-level $(l-1)$ participating peers send the video objects for them. The throughput of the system is contributed by the average fraction of full streaming of participating peer nodes. Some portion of the bandwidth is always used for chunk transfer to peers at the same level $(l-1)$ according to Figure 2. So we get the expression $P(l) = P(l-1) - N(l-1)$.

Dynamic cluster connections are presented in Figure 4. We observe that A, B, and C are the peer nodes at the level $(l-1)$ and D, E, F, and G are the peer nodes at the level (l). The dynamic clustering is a logical cluster construction among the peer nodes at every level for the real time requirement at any session. The peer nodes B and C at level $(l-1)$ exchange the chunk video objects or transfer among themselves. D is the cluster head at level (l) download video chunk from the dynamic cluster head "A" at level $(l-1)$. Now downloading chunk of video objects from the lower level gives priority over the "chunk video objective" transfers in the same level of multitier architecture in that session. Let us consider that N_l is the number of peer nodes at the level l. To approximate the value N_l by N, the processing is as follows:

$$\begin{aligned}
P(l) &= P(l-1) - N(l-1) \quad \text{for } 0 \prec \lambda \prec 1 \\
&= [P(l-2) - N(l-2)] - N(l-1) \\
&= P(l-2) - N(l-2) - N(l-1) \\
&= [P(l-3) - N(l-3)] - N(l-2) - N(l-1) \\
&= P(l-3) - N(l-3) - N(l-2) - N(l-1) \\
&= \cdots \\
&= \cdots \\
&= P(0) - N(0) - N(1) - N(2) - \cdots - N(l-1) \\
&= P(0) - N_c - \lambda N_c - \lambda^2 N_c - \cdots - \lambda^{l-1} N_c \\
&= N - N_c\left(1 + \lambda + \lambda^2 + \cdots + \lambda^{l-1}\right) \\
&\approx N - N_c \frac{1-\lambda^l}{1-\lambda}.
\end{aligned} \tag{6}$$

Finally, N_l is approximate as $N_l \approx N - N_c((1-\lambda^l)/(1-\lambda))$.

Let us assume that $p_{k_l}^{(i)}$ is the probability that at least k_l out of N_l number of peer nodes actively participate in the l level of the multitier architecture at ith session. Active means that the peer node has the required portion of encoded video file and remains live during the chunk video object transfer in that session.

So we get the expression

$$p_{k_l}^{(i)} = \sum_{k=k_l+1}^{N_l} \binom{N_l}{k} \rho^k (1-\rho)^{N_l-k}. \tag{7}$$

So during chunk transfer of video objects, the aggregate bit rates depend upon the active participation of the peer nodes at every level $\{l, l-1, l-2, l-3, \ldots, 0\}$. The distribution of B in the expression ((4) and (5)) can be obtained from the multistate Markov chain. $S_i L_j$ is the sequence of active participation of peer nodes to transfer the chunk of video objects at the session i in the jth level. Since the download capacity at any level depends upon the uploading amount of just the lower level for that session, expression ((4) and (5)) becomes

$$\begin{aligned}
p\left(B \le C_{(i)}^{\dagger}\right) &= p\left(S_i L_l, S_i L_{l-1}, S_i L_{l-2}, \ldots, S_i L_0\right) \\
&= p\left(\frac{S_i L_l}{S_i L_{l-1}}\right) p\left(S_i L_{l-1}, S_i L_{l-2}, \ldots, S_i L_0\right) \\
&= p\left(\frac{S_i L_l}{S_i L_{l-1}}\right) p\left(\frac{S_i L_{l-1}}{S_i L_{l-2}}\right) \cdots p\left(S_i L_0\right) \\
&= p_{k_l}^{(i)} p_{k_{l-1}}^{(i)} p_{k_{l-2}}^{(i)} \cdots p_{k_0}^{(i)} = \prod_{j=0}^{l} p_{k_{l-j}}^{(i)}.
\end{aligned} \tag{8}$$

Clearly, $p(S_i L_0) = 1$ for session like session i, by using expression (7), and after reordering the above sequence, we get

$$p\left(B \le C_{(i)}^{\dagger}\right) \approx \prod_{j=l}^{1} \sum_{k=k_j+1}^{N_j} \binom{N_j}{k} \rho^k (1-\rho)^{N_j-k}. \tag{9}$$

According to (5), we get

$$\prod_{j=l}^{1} \sum_{k=k_j+1}^{N_j} \binom{N_j}{k} \rho^k (1-\rho)^{N_j-k} \le (1-\eta). \tag{10}$$

Since N_j is the number of peer nodes at the jth level in multitier architecture, in a session k_m is the least required numbers of actively participating peer nodes at the mth level in mesh architecture during smooth transfer of chunk video objective. According to the expression, (10) holds true and inequality (1) becomes $k_l + k_{l-1} + k_{l-2} + \cdots + k_0 \leq k^*$:

$$\sum_{j=0}^{l} k_{l-j} \leq k^*, \tag{11}$$

where η is controlled by some threshold that depends on the adaptive bit rate for both sides of sender-receiver CODEC. The CODEC runs at the viewer's end and the content transferring peers or proxy server end. To find the optimized cost for transferring the chunk of the video objects, now the issue comes to efficiently finding the video object chunk storage with minimum hop counts in multitier architecture. To keep the value η at the lower level, the problem is to minimize the unnecessary searching packet forwarding and inter server gossiping. We propose the heuristic based path search for finding the chunk container of peer nodes for transferring video objects by using Algorithms 1 and 2. The estimated cost function is used to estimate the distance for possible destination of peer nodes in that particular session. The initial information about the higher-level peers is already available to the proxy server in multitier mesh architecture. For finding the peer, the search path is linear. So the cost of the path is optimized, that is, to avoid the unnecessary query forwarding to minimize the inter server gossiping. The path is considered the route to the farthest number in that session. The root is the proxy server. The distance is measured with the hop counts. In reality, more than one peer's node may contain the required chunk of video objects. So the search path will not be always linear. In the path search computation at a session, some extra additional link exists. So we can express it in form of the expression:

$$\begin{aligned}
&\text{Min} \quad \{\text{Max}\, d\,(\text{proxy server}, v) + \text{number of duplicate links exist in } S \mid \text{such that } v \in S, S \text{ is the Collection of churn storage peers.}\} \\
&\text{Subject to} \quad \prod_{j=l}^{1} \sum_{k=k_j+1}^{N_j} \binom{N_j}{k} \rho^k (1-\rho)^{N_j-k} \leq (1-\eta), \quad 0 \prec \rho \prec 1 \\
&\qquad \sum_{l=0}^{l} k_l \leq k^*, \\
&\qquad k_j \prec N_j, \\
&\qquad k^* \prec N.
\end{aligned} \tag{12}$$

To find the least-cost path that is minimum hop count towards the fastest peer storage is considered. The consisting hashing in the structured overlays for the multitier Peer to Peer network is also being used. The request for the portion of the chunk video object is forwarded through the longest prefix match via Algorithm 2. The consistent hash function [43] is assigned to each node and key by a fixed bit identifier using a base hash function. A node's identifier is chosen by hashing the node IP address, while a key identifier is produced by hashing the key. Here the used term "key" refers to both the original key and its image under the hash functions. Similarly, the term node refers to both the node and its identifier under the hash function. Consistent hashing is assigning key to node. In point to point network, name based consistent hashing maps key onto a node. Both keys and nodes are assigned with an m-bit identifier. For nodes, this identifier is a hash of the node's IP address. For keys, this identifier is a hash of a keyword, such as a file name or query string.

3.1. Proposition and Algorithm

Statement. The hop count between query emitter node for the chunk of video object x_s and query matched x^* node (that node has chunk of video objects) is at most $\|x_s - x^*\| \leq \log_k(n) \prec\prec (n-1)$ for $n \geq 2$. n is the size of multitier network and k is the base of the prefix for emitter node.

Proof. Let x_s be the query initiative node and let x^* be the query matched node. $(k-1)$ is the size of the adjutancy list for a node that has a base prefix k. When the query emits node the same as the query matched node, that is, proxy server, then inequality (13) holds true for the single hop neighbor, and $n \geq 2$:

$$\|x_s - x^*\| \leq \log_k n. \tag{13}$$

When the query is forwarded through the intermediate nodes $x_1, x_2, \ldots, x_m$ starting from the emitter node x_s to sink node x^*, the node x_s has the adjustable list size $(l_s - 1)$ with base of the prefix l_s, $x_{(1)}$ has adjustable list size $(l_1 - 1)$ with base of the prefix l_1, and so on; likewise, $x_{(m)}$ has the adjustable list size of $(l_m - 1)$ with the base of the prefix l_m. Query is forwarded by incremental prefix routing. So the path distance is expressed as

$$\begin{aligned}
\|x_s - x^*\| \leq \|x_s - x_{(1)}\| + \|x_{(1)} - x_{(2)}\| + \cdots \\
+ \|x_{(k)} - x^*\|.
\end{aligned} \tag{14}$$

The number of branches maintained at the node x_s is $(l_s - 1)$, the node $x_{(1)}$ maintains $(l_1 - 1)$, the number of branches, and so on. Now $x_{(m)}$ maintains $(l_m - 1)$, number of branches. As $x_{(1)} \in$ single hop neighbor of x_s, $x_{(i+1)} \in$ single hop neighbor of $x_{(i)}$ and $x^* \in$ single hop neighbor of $x_{(m)}$. It concludes $\|x_s - x_{(1)}\| \approx 1, \|x_{(i)} - x_{(i+1)}\| \approx 1, \ldots, \|x_{(k)} - x^*\| \approx 1$.

Expression (14) becomes $||x_s - x^*|| \prec (n-1)$. □

```
Input:   x_s // The Query emits node
         x* // The Query sink node
      Initialize
        Buffer1 ← finite size
        Hop_count (every node) ← (–1) // peers at
                                      // every level
        Begin
        Query initiated from x_s // Proxy server
        If {(x_s.node_id = =x*. node_id) && found (query massage)}
          then
          Hop_count = 0
          Return Hop_count
          Output: chunk present in proxy server
          Exit
          End
        else
for each x ∈ {single hop node from adjutancy list of x_s}
Compute: Query forward to the longest prefix match node x using Algorithm 2.
Buffer1 ← push (x)
while (Bufer1 non empty) do
{
  pop node (v) from Buffer1
    {
      for each w from N_b(v) // select all the
                    //single hop neighbor w of v
       {
        If  (Hop_count(w) == (–1)) // peer node yet
                          //not received the query
        then
        Hop_count (w) = Hop_count (v) +1;
    {
    If ((node identifier(w)==node identifier(x*))&& match (keyword_identifier))
     then // peer node having the query stream
     return Hop_count(w);
    Output: chunk present
      else w ∈ N_b(x*) // selected w belongs to
                    // single hop neighbor of x*
        {// heuristic search
        return
        Hop_count (w) +1;
        } // end search
    } // end block
    else
    Output: "Search query stream does not exist in this network".
    } // End for each
    } // End Pop
    } // End While
    End begin
    Stop
```

Algorithm 1: Heuristic Path Search Algorithm, query initiated from the proxy server.

Algorithms 1 and 2 are developed with the following consideration. The query message is generated from the proxy server. It is bits of a string header of the chunk video objects. There is more than one peer node present in the network. Those possess the required chunk video objects. The aim of the algorithm is to find the fastest least-cost path towards the desired peer node. The query message is routed to desire peer node via key identification.

```
Variable:
Prefix: list
L: Array of Record
LPM, M: Record // LPM is longest prefix match
D, D_l, D': Array of Character
R, i, l, A: integer
Initialize
Prefix ← {set of peer nodes}
L [i].length ← {Prefix length of ith distinct length}
L [i].hash ← {Hash table}
D ← {Storage server address}
B ← {Range of the array}// 32 bits for IPv4, 128
                        // bits for IPv6
M ← {Binary string of size R}
Begin
 A = 0;
   while (R ≠ NULL) do
   {
      i = ⌊(A + B)/2⌋;
      l = L[i].length; // longest possible length
      D_l = [(Bit String of D) base 2] ≤ l;
      D' = Assign (Most significant bit of D_l);
      // Index position of the binary string associated
      // to D' in hash table.
      M = Search (D', L [i].hash);
      // search hash table for D'
         If (M = = Null) then B= ⌊(A + B)/2⌋ − 1;
         // search in lower half
         else A = ⌊(A + B)/2⌋; // search in upper half
         if (M ∈ LPM) then break; else continue;
   }// end while
Update: Query forward to (M.address)
End begin
Stop
```

Algorithm 2: Query forward to longest prefix match

4. Simulation Results and Discussion

The storage server contains the raw file (or coded video). The YUV file is created by decoding the raw file. At first, it is converted to ∗.m4v file format. Then, we create the mp4 container and simultaneously create the reference videos. The frames are packetized for transport over the network. Here mp4trace command sends the ∗.mp4 file to the destination peer node through the freely available port except the system port (usually selected ports above 5000) [44]. The transportation of these chunks of video objects, that is, "the sequence of packet frames," is done with RTP/UDP. All this design and implementation of EvalVid-RA, a tool set for rate adaptive VBR (video bit rate) that is added to ns-2, are based on some modification to the EvalVid Version 1.2 tool and the ns-2 interfacing code. The ffmpeg command of EvalVid is used to control rate of some GOP before stabilizing the rate output. The VBR rate controller is set to 600 kbit/s for the individual session. The mesh shaped hybrid multitier architecture, including the cache proxy server, has 10 levels. The higher level l_9 is the cache proxy server and l_0 level is for the video storage server. Levels l_1 to l_8 possess the peer nodes in mesh shaped multitier. Each level has 4 peers that are interconnected. The peers that belong to the same level exchange buffer information to transfer the chunk video objects. The peers at the higher level are connected to lower-level peers in unicast; that is, the higher-level peers only download the chunk of video objects from the lower level. In the environment, we consider two types of simulation. In type one, the peers maintain dynamic connection. In dynamic connection, every peer has adjacent list size of 4 and 6. These links are selected from the available 11 links to the peer. The 11 links are orientated as 4 links for the unicast downloading from the lower-level peers, 4 links for the unicast uploading for the higher-level peers, and 3 links for the chunk transfer at the same level of peers. In the second type of simulation, we consider the static links 4 and 6 out of the 11 available links to the peer. In the static link, full content of the video is transferred from the storage server or the intermediate peers through the initial fixed setup links for every session. The link capacity maintains 400 kbps to 800 kbps speed. 400 kbp is the link capacity between the viewers to the cache proxy server. The 800 kbps is the link capacity of the storage server to the next higher server (i.e., billing server). The intermediate-level mesh peers maintain the link capacity of 600 kbps. The required parameter values and ranges are summarized in Table 1.

Table 1: Simulation parameters.

Parameter	Values/range
Number of levels in multitier	10
Number of peers at each level	4
Raw video frame rate	30/second
Group of picture length	30 frames
Link capacity	400 kbps to 800 kbps
Number of viewers connected to each proxy	20 (Min)
Number of viewers connected to each proxy	40 (Max)

The confidence interval used in this simulation is 90%. For small prototype evaluation, we restricted the number of viewers connected to proxy server and adjacent list size of peer node. The viewers are connected to each proxy server; minimum is 20 and maximum is 40. The number of viewers connected to the proxy server can grow. The sum-up link capacity of the viewers connected to the proxy server is more than the intermediate link capacity.

The simulation result is based on the request initiative from the proxy server, when the viewers submitted his/her requests through the proxy server. In plotting Figures 5–10, we consider the vertical axis as the number of requests sent (cumulative) for the chunk video objects from the cache proxy server. The horizontal axis is the number of hops to the location of those chunk video objects. The request initiative from the cache proxy server is approximately equal to $100999 \approx 10^6 + 10^3 + 10^1$.

Figures 5 and 7 present the simulation result of minimum hop count to the required chunk of the video objects towards the peers or storage server. The peer nodes at ith level,

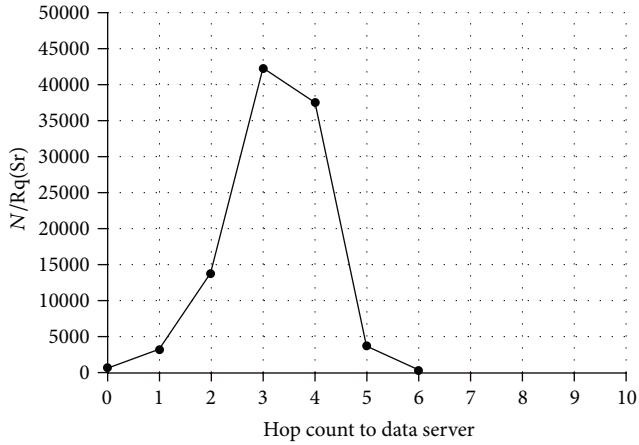

Figure 5: Dynamic adjacent list size (4).

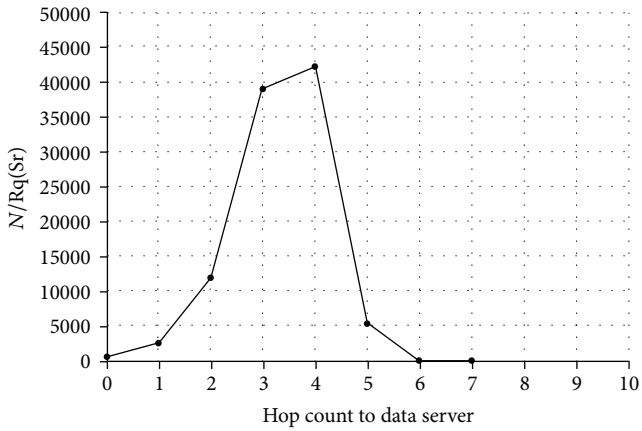

Figure 6: Static adjacent list size (4).

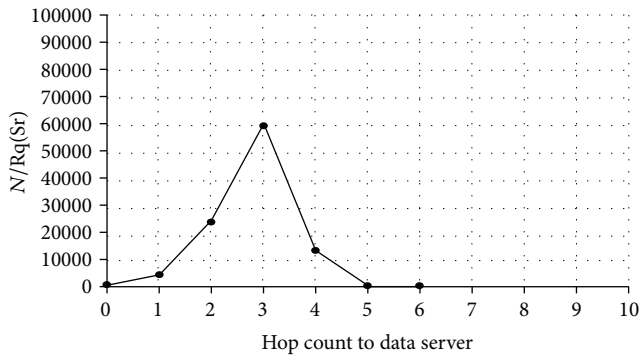

Figure 7: Dynamic adjacent list size (6).

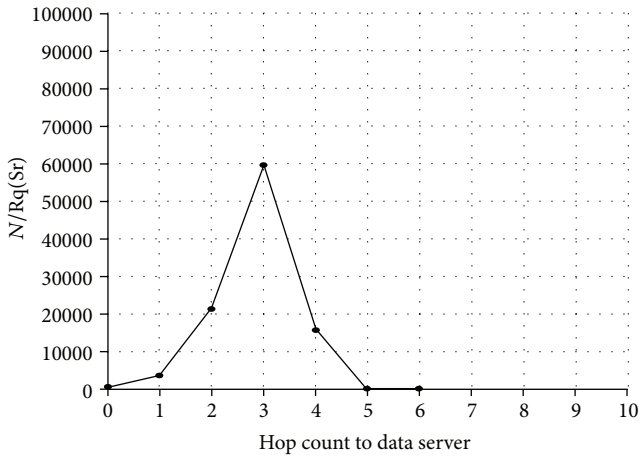

Figure 8: Static adjacent list size (6).

for $0 \leq i \leq 9$, maintain dynamic adjacent list sizes 1 to 4 and 1 to 6. Figures 6 and 8 present the simulation result where every peer at every level maintains static adjacent list sizes 4 and 6. To search for destination peers address, we simply start with the longest length hash table and extract the first bits and do a search in the hash table for length entries. If we succeed, we have found the longest prefix match and thus our BMP (best match path); if not, we look at the first length smaller than previous match path (done by the array indexing one position less than the position) and continue the search. The search is forwarded by incremental prefix routing. Keeping $(b - 1)$ hosts at each prefix digit b is the base of the prefix. To minimize the hop count, we consider the number of links at each intermediate node dynamically changing from one to $(b - 1)$. So expression (12) will be always bounded by $\log_b(N)$, and N is the number of "video objects storage" peer nodes. So $\log_b(N)$ is the maximum hop to any destination. The comparative simulation results are presented in Figures 9 and 10. The simulation result shows that 43000 numbers of requests were served from the peer at the hop count 3 for the dynamic adjacent list size 4. We have noticed that 40000 numbers of requests were served from the peer at the hop count 3 for static adjacent list size 4.

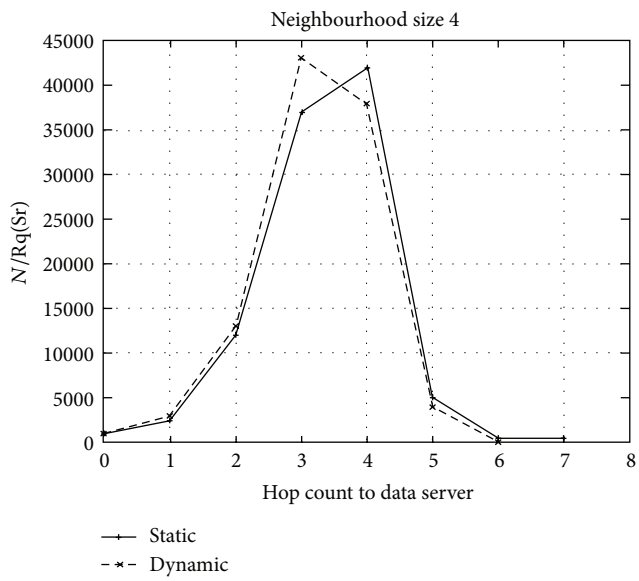

Figure 9: Compare dynamic versus static adjacent list size (4).

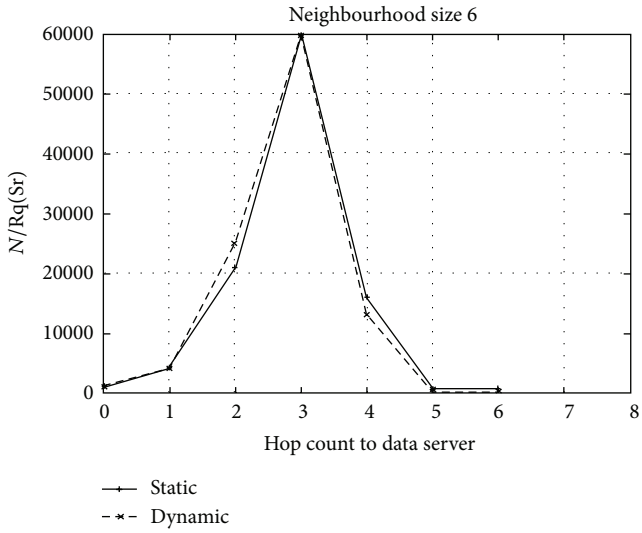

Figure 10: Compare dynamic versus static adjacent list size (6).

The chunk size is the same for both the cases. In Figures 9 and 10, we have noticed that the dotted line (dynamic) lies above the nondotted line (static) between the hop count 0 and hop count 3. In the next phase, the nondotted link lies above the dotted line between the hop counts 3 and 7. The longest prefix match with the dynamic adjacent list on the basis of "Heuristic Path Search Algorithm" gives better cost-effective results.

The simulation results with the dynamic adjacent list size 4 and static adjacent list size 4 are presented in Figure 9. The simulation result clearly shows that, in dynamic environment with list adjacent size 4, 35×10^2 numbers of video objects serve from the hop count 1. 130×10^2 numbers of video objects serve from hop count 2. 430×10^2 numbers of video objects serve from hop count 3. 370×10^2 numbers of video objects serve from hop count 4. 40×10^2 numbers of video objects serve from hop count 5. Static environment shows that, with adjacent list size 4, 30×10^2 numbers of video objects serve from hop count 1. 120×10^2 numbers of video objects serve from hop count 2. 360×10^2 numbers of video objects serve from hop count 3. 410×10^2 numbers of video objects serve from hop count 4 and 50×10^2 numbers of video objects serve from hop count 5. Here, we consider a rough estimate to measure the cost of video object transfer by using the metrics, such as "number of video objects serves" and hop counts. Cost is the sum of the "number of video objects serves a_m" multiplied by the "hop counts b_m towards the container that possesses the video objects"; that is, $\sum_{m=1}^{n}(a_m * b_m)$. Here m is the hop count towards the chunk video container peer node, and n is an end of hop counts towards the container. The above expression gives the following cost score. Dynamic environment of simulation produced the score 3265×10^2 according to Figure 9. For static environment of simulation, the produced cost score is 3740×10^2 according to Figure 9. Similarly according to simulation the result is presented in Figure 10. That cost score in dynamic environment with adjacent list size 4 is 2840×10^2. The cost score in static environment with adjacent list size 6 is 2880×10^2. The cost score directly depends on the number of computations. The performance of the system increases as the score value decreases, during any stage of transfer of chunk video object through the multitier architecture.

5. Conclusions

In this paper, we have shown that the hybrid architecture of Peer to Peer network with mesh type's network enhances the content delivery of video objects. The number of requests generated by the proxy server serves from the distributed chunk video object storage at the various levels of the multitier architecture. The simulation used the static as well as the dynamic growth links at each middle tire containing storage nodes. The result showed the effectiveness of the hybrid architecture over the pure Peer to Peer network. The previous works considered only pure P2P network. The hybrid type's work successfully reduces the cost for the search path; hence it reduces the delay. Another aspect of the work is that it selects the required position of the video objects and supplies it to the desired node in the hybrid multitier architecture, which is the enhancement of previous Peer to Peer type stream data transmission. There is enough scope for further improvement.

Conflict of Interests

The authors Soumen Kanrar and Niranjan Kumar Mandal declare that there is no conflict of interests regarding the publication of the paper.

Acknowledgment

The authors are grateful to Sharmista Das Kanrar from Bishop Westcott, Ranchi, India.

References

[1] H. Jiang, J. Li, Z. Li, and X. Bai, "Effcient large-scale Content Distribution with Combination of CDN and P2P Networks," *International Journal of Hybrid Information Technology*, vol. 2, no. 2, 2009.

[2] C. Liang, Z. Fu, Y. Liu, and C. W. Wu, "Incentivized peer-assisted streaming for on-demand services," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 9, pp. 1354–1367, 2010.

[3] X. Hei, Y. Liu, and K. Ross, "IPTV over P2P streaming networks: the mesh-pull approach," *IEEE Communications Magazine*, vol. 46, no. 2, pp. 86–92, 2008.

[4] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross, "A measurement study of a large-scale P2P IPTV system," *IEEE Transactions on Multimedia*, vol. 9, no. 8, pp. 1672–1687, 2007.

[5] S. Kanrar, "Analysis and implementation of the large scale video-on-demand system," *International Journal of Applied Information Systems*, vol. 1, no. 4, 2012.

[6] H. Yin, X. Liu, T. Zhan et al., "Design and deployment of a hybrid CDN-P2P system for live video streaming: experiences with LiveSky," in *Proceedings of the 17th ACM International Conference on Multimedia (MM '09)*, pp. 25–34, ACM, Beijing, China, October 2009.

[7] W.-P. K. Yiu, X. Jin, and S.-H. G. Chan, "VMesh: distributed segment storage for peer-to-peer interactive video streaming," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 9, pp. 1717–1731, 2007.

[8] ITU-T Study Groups, "Definition of Quality of Experience (QoE)," ITU-T SG12, COM12-LS 62-E, TD 109rev2(PLEN/12), Geneva, Switzerland, January 2007, https://www.itu.int/md/T05-FG.IPTV-IL-0050/en.

[9] N. Zong, "Survey and Gap analysis for HTTP streaming standards and implementations," 2010.

[10] Q. Wu, "Problem statement for HTTP streaming," 2010.

[11] K. D. Singh, Y. Hadjadj-Aoul, and G. Rubino, "Quality of experience estimation for adaptive HTTP/TCP video streaming using H.264/AVC," in *Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC '12)*, pp. 127–131, Las Vegas, Nev, USA, January 2012.

[12] N. Goel, B. Raman, and I. Gupta, "Chaos based joint compression and encryption framework for end-to-end communication systems," *Advances in Multimedia*, vol. 2014, Article ID 910106, 10 pages, 2014.

[13] W. Zeng and S. Lei, "Efficient frequency domain selective scrambling of digital video," *IEEE Transactions on Multimedia*, vol. 5, no. 1, pp. 118–129, 2003.

[14] R. Draves, J. Padhye, and B. Zill, "Comparison of routing metrics for static multi-hop wireless networks," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 133–144, 2004.

[15] S. Banerjee and A. Misra, "Minimum energy paths for reliable communication in multi-hop wireless networks," in *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc '02)*, pp. 146–156, ACM, Lausanne, Switzerland, June 2002.

[16] S. Kanrar, N. K. Mandal, and S. D. Kanrar, "Session based storage finding in video on demand system," in *Proceedings of the 3rd International Symposium on Women in Computing and Informatics (WCI '15)*, pp. 131–135, Kochi, India, August 2015.

[17] A. Erramilli, O. Narayan, and W. Willinger, "Experimental queueing analysis with long-range dependent packet traffic," *IEEE/ACM Transactions on Networking*, vol. 4, no. 2, pp. 209–223, 1996.

[18] B. Maglaris, D. Anastassiou, P. Sen, G. Karlsson, and J. D. Robbins, "Performance models of statistical multiplexing in packet video communications," *IEEE Transactions on Communications*, vol. 36, no. 7, pp. 834–844, 1988.

[19] V. S. Frost and B. Melamed, "Traffic modeling for telecommunications networks," *IEEE Communication Magazine*, vol. 32, no. 3, pp. 70–81, 1994.

[20] J. Wen and J. D. Villasenor, "Reversible variable length codes for efficient and robust image and video coding," in *Proceedings of the Data Compression Conference (DCC '98)*, pp. 471–480, IEEE, Snowbird, Utah, USA, April 1998.

[21] F. Liu and H. Koenig, "Puzzle—an efficient, compression independent video encryption algorithm," *Multimedia Tools and Applications*, vol. 73, no. 2, pp. 715–735, 2012.

[22] E. Mallika and K. Sivakumar, "Joint video compression and encryption using secure wavelet transformation and entropy coding," *International Journal of Computer Science and Mobile Computing*, vol. 3, no. 2, pp. 483–489, 2014.

[23] C.-P. Wu and C.-C. J. Kuo, "Design of integrated multimedia compression and encryption systems," *IEEE Transactions on Multimedia*, vol. 7, no. 5, pp. 828–839, 2005.

[24] C. Kim and J.-N. Hwang, "Fast and automatic video object segmentation and tracking for content-based applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 2, pp. 122–129, 2002.

[25] K. J. Singh and R. Manimegalai, "A survey on joint compression and encryption techniques for video data," *Journal of Computer Science*, vol. 8, no. 5, pp. 731–736, 2012.

[26] M. M. Hannuksela, D. Rusanovskyy, W. Su et al., "Multiview-video-plus-depth coding based on the advanced video coding standard," *IEEE Transactions on Image Processing*, vol. 22, no. 9, pp. 3449–3458, 2013.

[27] T. Wiegand and G. J. Sullivan, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[28] F. H. P. Fitzek and M. Reisslein, "MPEG-4 and H.263 video traces for network performance evaluation," *IEEE Network*, vol. 15, no. 6, pp. 40–54, 2002.

[29] D. A. Tan, K. A. Hua, and T. Do, "ZIGZAG: an efficient peer-to-peer scheme for media streaming," in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communication (INFOCOM '03)*, vol. 2, pp. 1283–1292, San Francisco, Calif, USA, April 2003.

[30] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '02)*, pp. 205–217, 2002.

[31] Y. Liu, Y. Guo, and C. Liang, "A survey on peer-to-peer video streaming systems," *Peer-to-Peer Networking and Applications*, vol. 1, no. 1, pp. 18–28, 2008.

[32] S. Kanrar and N. K. Mandal, "Dynamic page replacement at the cache memory for the video on demand server," in *Advanced Computing, Networking and Informatics-Volume 2: Wireless Networks and Security Proceedings of the Second International Conference on Advanced Computing, Networking and Informatics (ICACNI-2014)*, vol. 28 of *Smart Innovation, Systems and Technologies*, pp. 461–469, Springer, Berlin, Germany, 2014.

[33] S. Kanrar and N. K. Mandal, "Optimum storage finding in video on demand system," in *Proceedings of the 2nd International Conference on Signal Processing and Integrated Networks (SPIN '15)*, pp. 827–830, IEEE, Noida, India, Feburary 2015.

[34] N. Leibowitz, C. A. Bergman, R. Ben-shaul, and C. A. Shavit, "Are files wrapping network scacheable? Characterizing P2P traffic," in *Proceedings of the 7th International WWW Caching Workshop*, August 2002.

[35] C. Vincenzo, R. Gaeta, R. Loti, and L. Liquori, "Inter connection of large scale unstructured p2p networks: modeling and analysis," in *Analytical and Stochastic Modeling Techniques and Applications*, vol. 7984, pp. 183–197, Springer, 2013.

[36] A. Brocco and I. Baumgart, "A framework for a comprehensive evaluation of ant-inspired peer-to-peer protocols," in *Proceedings of the 20th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP '12)*, pp. 303–310, IEEE, Garching bei München, Germany, February 2012.

[37] J.-S. Li and C.-H. Chao, "An efficient superpeer overlay construction and broadcasting scheme based on perfect difference graph," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 5, pp. 594–606, 2010.

[38] R. Rodrigues and P. Druschel, "Peer-to-peer systems," *Communications of the ACM*, vol. 53, no. 10, pp. 72–82, 2010.

[39] S. Kanrar, "Performance of distributed video on demand system for multirate traffic," in *Proceedings of the International Conference on Recent Trends in Information Systems (ReTIS '11)*, pp. 52–56, Kolkata, India, December 2011.

[40] S. Kanrar and N. K. Mandal, "Performance enhancement for audio-video proxy server," in *Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014*, vol. 327 of *Advances in Intelligent Systems and Computing*, pp. 605–613, Springer, Basel, Switzerland, 2015.

[41] F. Peng, A. Malatras, B. Hirsbrunner, and M. Courant, "Constructing multi-layer overlays for pervasive environments," in *Proceedings of the 8th International Workshop on Mobile P2P Computing (MP2P '12)*, pp. 1–6, March 2012.

[42] S. Tewari and L. Kleinrock, "On fairness, optimal download performance and proportional replication in peer-to-peer networks," in *Proceedings of the 4th International IFIP-TC6 Networking Conference*, Waterloo, Canada, May 2005.

[43] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup service for internet applications," in *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '01)*, pp. 149–160, ACM, San Diego, Calif, USA, August 2001.

[44] B. Tan and L. Massoulie, "Optimal content placement for peer-to-peer video-on-demand systems," *IEEE/ACM Transactions on Networking*, vol. 21, no. 2, pp. 566–579, 2013.