Peer-to-Peer Networking and Applications (2021) 14:559–584
https://doi.org/10.1007/s12083-020-01002-4

A reliable peer-to-peer streaming protocol in low-capacity networks

Oluwafolake E. Ojo[1] · Ayodeji O. Oluwatope[2] · Suraju O. Ajadi[3]

Received: 13 April 2020 / Accepted: 8 September 2020 / Published online: 14 October 2020

Abstract
The recent global demand for video streaming applications has paved the way for peer-to-peer streaming system (P2PSS). Strategic scheduling scheme and dynamic overlay topology are essential to maintain quality of service (QoS) and quality of experience (QoE) in P2PSS. The concept of P2PSS was tailored towards relying on active peers' bandwidth to achieve cheap and scalable means of distribution over the Internet, such that peers with highest bandwidth serve as backbones for others. However, selecting backbone peers in low-capacity network environment is challenging due to insufficient bandwidth and poor infrastructure, thereby resulting in poor QoS and unpleasant user's QoE. In this paper, we conducted a survey on users' experiences with live video in selected locations in Nigeria. We designed an adaptive P2P streaming protocol and performed a packet-level simulation in Network Simulator 3(NS-3). Diverse simulation scenarios were set up to evaluate the proposed streaming protocol. Trace files data were analysed to measure end-to-end delay, start-up delay, and throughput. Furthermore, the proposed streaming protocol was benchmarked against selected existing schemes. The evaluation results revealed a 7.4% and 28% reduction in start-up and in end-to-end delays and 9% increase in throughput.

Keywords Peer-to-peer networks · Video streaming · Quality of Service · Quality of experience · Low-capacity networks

1 Introduction

The evolution of the Internet has enabled the rapid growth of multimedia systems ranging from simple music downloading to watching television over the Internet [1]. Research has revealed that, over the last two decades, watching/streaming video applications are the most popular on-line activities [2, 3]. For instance, the popular Netflix is reportedly streaming over 1 billion hours of video per month which is equivalent to about 7,200,000 Tera bytes of video traffic, with this figure rising constantly [4]. The forecast that 78% of global multimedia transmission across Internet will be video based by 2021 [5] serves as a motivation for systematic investigations into video streaming techniques [6–10]. According to literature, data transmission and routing between nodes are two key challenging issues in data networking especially in low-powered devices such as Internet of Things (IoTs) [11].

Network architecture is a major determinant in ensuring successful transmission of video applications from source to destinations across the Internet. Recently, it has been perceived more as a framework which specifies not only network topology, network type, network components and their functionalities, but as well as data communication protocols, data formats and supported services [12]. For instance, the most popular multimedia streaming protocols are real-time transport Protocol (RTP) and real time control protocol (RTCP). The RTP relies on lower-layer services while RTCP is a companion protocol to RTP which feedbacks QoS statistics from the receiver to the sender [13]. Researchers have also applied micro-protocol to address unreliability problem in Voice over Internet Protocol steganography by embedding data over covert storage channels or covert hybrid channels where RTP serves as underlying protocol [14].

✉ Oluwafolake E. Ojo
ojoeo@funaab.edu.ng

Ayodeji O. Oluwatope
aoluwato@oauife.edu.ng

Suraju O. Ajadi
sajadi@oauife.edu.ng

[1] Department of Computer Science, Federal University of Agriculture, Abeokuta, Nigeria

[2] Department of Computer Science and Engineering, Obafemi Awolowo University, Ile-Ife, Nigeria

[3] Department of Mathematics, Obafemi Awolowo University, Ile-Ife, Nigeria

Generally, network architectures are grouped into centralized, decentralized and distributed networks as shown in Fig. 1. Most networks today are distributed, wherein tasks are sub-divided among multiple nodes. In distributed networks, separate nodes handle a subset of the tasks instead of one single large machine being responsible for all aspects of a process [15]. The most commonly distributed network architectures are client-to-server (CS), content distribution network (CDN) and peer-to-peer (P2P) architectures.

It was observed that the CS architecture is prone to failure in the event of numerous users requesting video files from a common central server at the same time [16]. This constraint is major, and therefore renders CS unsuitable for effective delivery of video contents in large scale content distribution. Although, the CDN architecture has proven to improve on the shortcoming in CS with the introduction of multiple servers [17], such that video files are distributed to multiple servers at strategic locations closest to the end-users [18]. This feature enables the CDN architecture to effectively provide large-scale video-on-demand and live video streaming services to a large number of Internet users [19]. However, the cost of providing a robust CDN video streaming system; such as YouTube [20] and Netflix [21], is a limiting factor. In addition, another shortcoming of the CDN is its sole reliance on the CDN servers ignoring user's upload bandwidth, thereby, transferring the bandwidth burden to the CDN infrastructure [17].

The P2P architecture suffices as a more dependable technique for disseminating video content across the Internet. The reason being that it does not rely on dedicated delivery infrastructure and hence, offers the possibility of rapid deployment at low cost [22]. P2P systems have been largely successful in large scale content distribution [23, 24], particularly for video streaming services due to its scalability feature [25]. Leveraging the scalability of P2P networks in media streaming is a well-studied area in the academic literature [26]. Aside from scalability, the P2P architecture exhibits decentralization, self-organization, fault resilience, ad hoc connectivity, and low construction costs [27]. P2P architecture also guides against network failures as well as enabling a large number of users to access the network with relatively low resources. P2PSS relies on peer cooperation in which case peers are expected to contribute upload bandwidth and redistribute video contents from one peer to another [28].

Network topology is yet another important component in network architecture. In which case, it describes the representation of the interconnectedness among nodes in the network. This representation exists at three different levels: link, Internet and overlay topologies [29]. The most popular network topologies are ring, star, bus, tree and mesh. In a ring topology, each node is connected to two other nodes, forming a single data path in form of a ring. Each host in a star topology is connected to a central switching facility. In bus topology, a common backbone link connects all the nodes one to the other. However, the tree topology consists of a combination of bus and star topologies. Also, each host is directly connected to any other host within that network in a mesh network topology [12]. In P2P streaming systems, the overlay topology is classified as - tree-based (TB) and mesh-based (MB) as shown in Fig. 1 [30].

Network speed is another component that determines smooth transmission of video applications to end-users. Research has shown that the network speed experienced

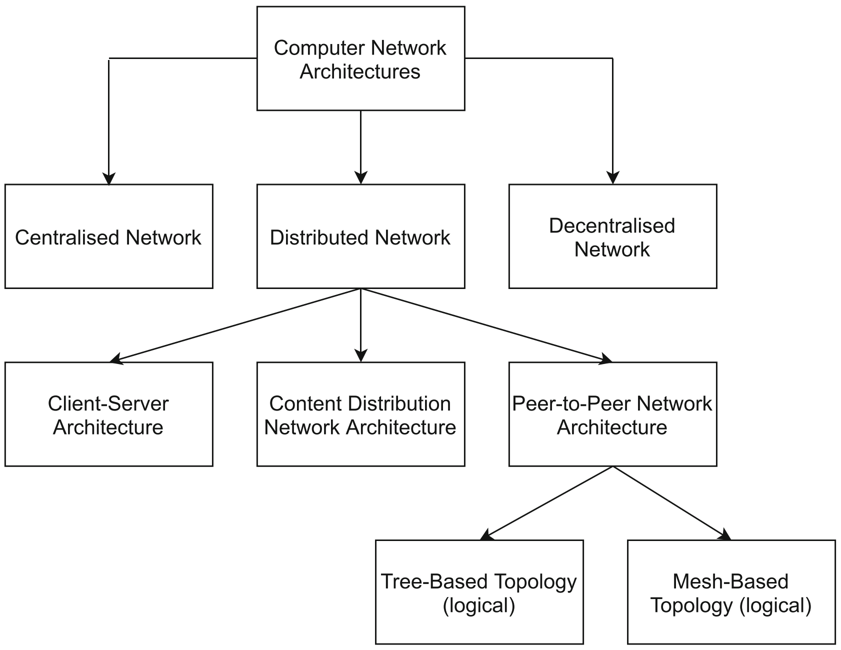

Computer Network Architectures
Centralised Network
Distributed Network
Decentralised Network
Client-Server Architecture
Content Distribution Network Architecture
Peer-to-Peer Network Architecture
Tree-Based Topology (logical)
Mesh-Based Topology (logical)

Fig. 1 Organogram of computer network architecture

by Internet users in developing countries is quite different from what is obtainable in most developed countries [31]. For example, the United State Federal Communication commission(US,FCC) in 2015 upgraded its minimum upload speed from 4 megabits per second(Mbps) to 25Mbps and minimum download speed from 1 Mbps to 3 Mbps (25/3 Mbps) [32]. In addition, the recent report of US, FCC showed that vast majority of Americans; above 85% now have access to fixed terrestrial broadband service at 250/25 Mbps [33]. Unfortunately, the situation is not the case in developing countries in terms of obtainable upload/download speeds. The case of Nigeria being a good reference [34]. It is noteworthy to mention that Internet penetration in Nigeria is the highest in Africa, however, Nigerian experiences a low video streaming uptake due to slow download speed and inflexible data plan [35]. Furthermore, our recent survey on campus networks in Nigeria revealed limited bandwidth, poor load balancing policy across critical equipment and poor Infrastructure as being the main challenges to achieving state-of-art video transmission speed. Therefore, we define a low capacity network as a network with speed $< 25/3$ Mbps.

Despite the wide deployment of P2PSS over the Internet, long delay, high packet loss and unplanned interruptions are barriers to realising optimised overlay topology for video streaming in low-capacity network. Furthermore, heterogeneity of peers, flash crowd, high churning situation, uneven distribution of traffics and resource assignment based on locality are some of the issues with the current overlay topology researchers are dealing with. In order to ensure video streaming infrastructure guarantees minimal packet delay and packet loss, it is necessary to design a robust overlay topology capable of curtailing large variations in network response as the P2P network scales. This paper documents the packet-level simulation of the re-engineered UStream earlier presented by [36]. We have thoroughly investigated the chaotic dynamics of UStream [37] and its behavioural pattern [38], hence, we present a reliable streaming protocol suitable in low-capacity networks.

Our main contributions in this paper are:

(i) we proposed an effective scheduling mechanism for smooth transmitting of live videos to improve QoS in P2P network;
(ii) we constructed and formalised a hybrid overlay topology for live video streaming in order to minimise peer starvation, node failure, peer churn and flash crowd; and
(iii) we introduced the concept of approximate peers to overlay topology such that incoming peers are connected to single parent or super node based on request and similar attributes. Specifically, we introduced the idea of alternate parent peers as substitute peers to the parent peers in case of node failure, thereby, enhancing playback continuity and significantly reduces packet loss.

The rest of the the paper is organized thus: Section 2 gives background knowledge on P2PSS; Section 3 discusses the existing works in P2P streaming topologies and investigation on user's QoE with streaming services in low capacity networks, Section 4 presents the proposed streaming protocols conceptualisation and formalisation, Section 5 presents algorithms and complexities; Section 6 presents performance evaluation including experiment setup, simulation results and discussions; and Section 7 concludes this paper and discusses potential future work.

2 Peer-to-peer streaming system

In P2P streaming, a server only needs to stream to some users, who in turn share the stream received with neighbours. The main advantage of P2P streaming systems lies in resource distribution using user's uplink bandwidth which leads to lowering of bandwidth requirement at the server. P2P streaming system has been shown to be effective in serving quite a large group [39]. P2P multimedia streaming from one or multiple multimedia sources (that is, streaming servers) to a group of participating peers helps to relieve the bandwidth cost burden on the server. If the individual peers contribute as much bandwidth as they consume, the streaming session is scalable to a large number of peers in the session [40]. P2P multimedia streaming applications need to reduce network traffic to address Internet Service Provider's concerns without sacrificing the quality of users viewing experience [25]. P2P streaming systems can be broadly classified into two categories based on the overlay network structure namely tree-based and mesh-based [16, 30, 41]. Tree-based P2P streaming system, as shown in Fig. 2, is an extension of single-tree multi-cast routing in which one overlay routing tree rooted at the server is constructed and maintained centrally [42]. The tree-based approach explicitly places peers in a single tree or multiple multi-cast trees, where they receive the stream from their parent(s) and forward it to their children [43]. Most tree-based approaches are typically push-based, that is, when a node receives a data packet, it also forwards copies of the packet to each of its children [44]. Each peer in tree-based topology determines an appropriate number of trees to join based on its access link bandwidth and is able to receive data only from its specified parent node. In a situation where a parent node fails or leaves the network, its whole sub-tree loses data until the tree is reconstructed. Due to the structured nature of tree-based topology, the impact of high

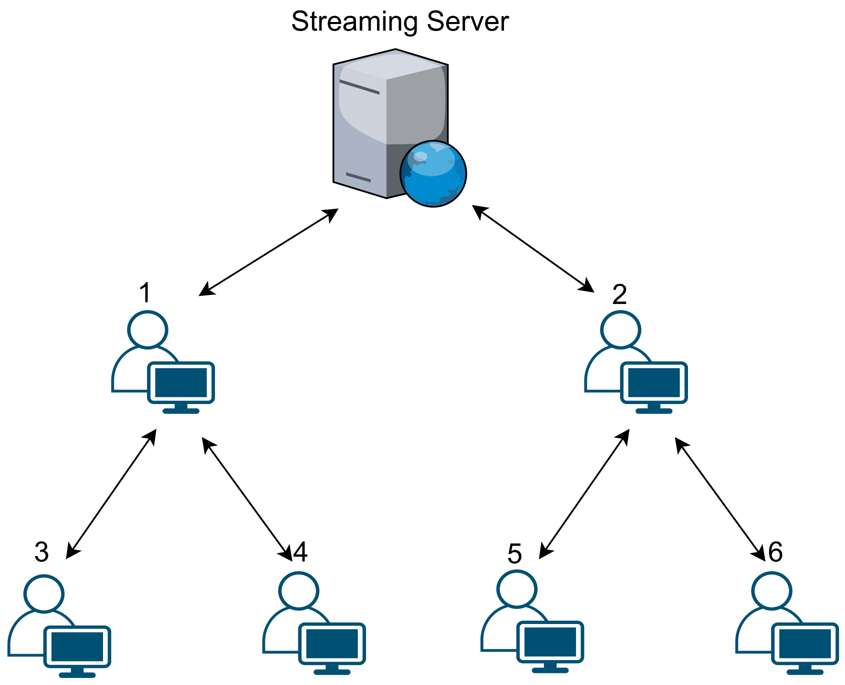


Fig. 2 Tree based topology [45]

rate of churning, the performance of P2P media streaming system is impaired [45]. A tree can always accept a new internal node. However, in the presence of churn, a tree could become saturated, and thus unable to accept any new leaf node [46].

Multi-tree topology is an unstructured topology; which mean this type of topology does not use distributing hash table, in which there are more than one sub-tree instead of one streaming tree [47]. In multi-tree streaming as shown in Fig. 3, the server divides the stream into multiple sub-streams. Instead of one streaming tree, multiple sub-trees are constructed, one for each sub-stream. Each peer joins all sub-trees to retrieve sub-streams, within each sub-tree, the corresponding sub-stream flows down level by level from the source server to all the leaf nodes. A peer has different positions in different sub-trees. It might be positioned on an internal node in one subtree and on a leaf node in another subtree [48]. Multi-tree topology can be considered as a combination between the simplicity of tree topology and unstructured topologies. This topology has two drawbacks: The first is increasing the overhead of the streaming compared to tree topology. The second occurs when a peer becomes a leaf in all sub-trees and contributes only in downloading without uploading [47].

In the mesh-based approach as depicted in Fig. 4, participating peers contribute their resources (i.e., outgoing bandwidth) more effectively. This, in turn, improves the utilization of available resources among peers and leads to a better scaling property for mesh-based approach compared to the traditional tree-based approach [49]. Mesh topologies can be unstructured or structured. In unstructured meshes, peers are connected to randomly chosen peers to provide more neighbours and different delivery paths. This leads to robustness when failures occur, since every piece can be obtained from other peers in a simple way. In structured meshes, on the other hand, peers are typically arranged into clusters, with the majority of links being established within the same cluster [50]. Generally, mesh-based architectures adopt a swarm-like design, where a peer periodically exchanges data availability information with its neighbouring peers, pro actively pulling packets from other peers that have the requested data, and pushing packets to other peers that are expecting the data [51]. The major advantages of mesh overlays in comparison to the tree-based overlay are the simple design principle and inherent robustness, particularly desirable for the highly dynamic, high-churn P2P environment [52].

2.1 Quality of service in peer-to-peer streaming

In general, quality of service (QoS) has three attributes to measure the output performance of a process: timeliness (delay and jitter), preciseness (required bandwidth or throughput) and accuracy (error rate) [53]. A brief review of QoS evaluation in video streaming either VoD or live streaming is given as follows:

2.1.1 Packet delay and start-up delay

Video streaming delay is a common factor in transmission of video streams in P2P networks. The various possible paths of video streams in the form of video packet may

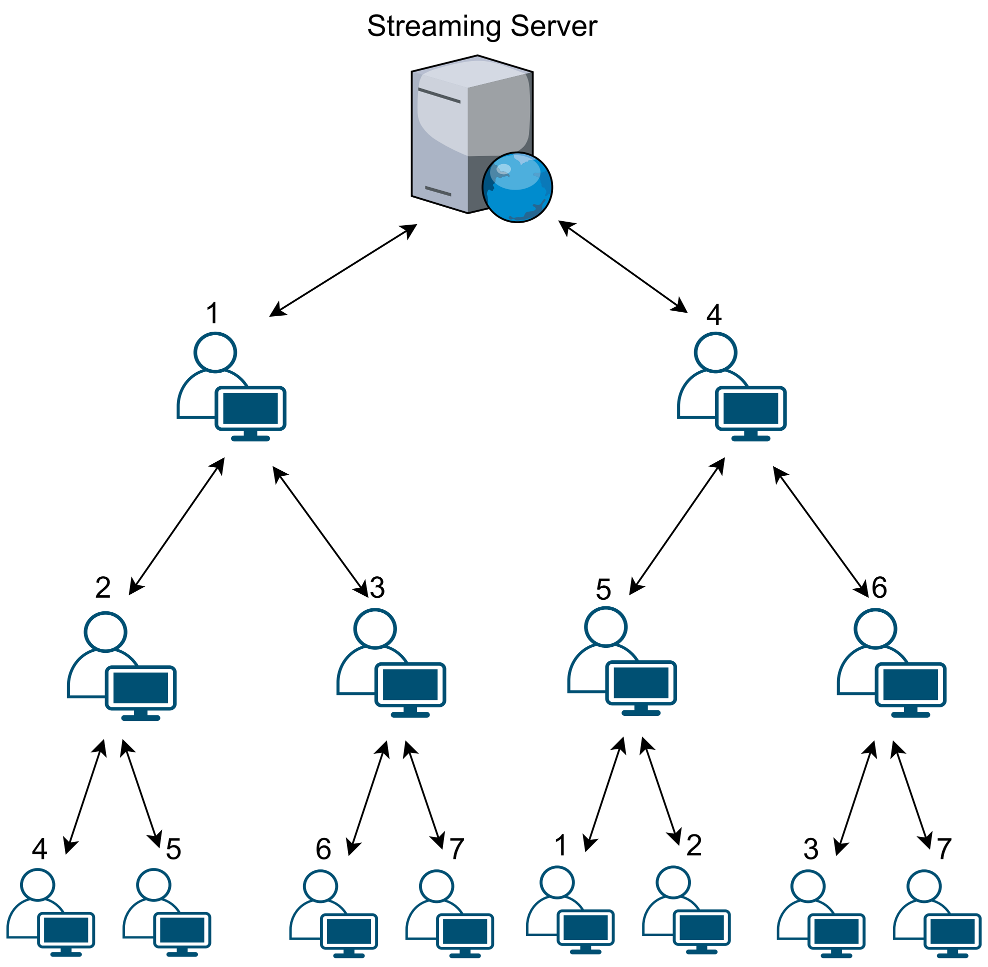


Fig. 3 Multi-tree topology [48]

have to travel as well as other factors like hardware, rate, bandwidth and congestion along the different paths can lead to delay in the arrival of video streams among the peers. Usually, video transmission protocols handle the arrival of delayed video streams through buffering. It becomes necessary to compute the end-to-end delay for real time streaming applications in order to ensure robustness and guaranteed bandwidth [54]. According to [55], the end-to-end delay can be estimated for transmission of video streams between any two Peer$_i$ and Peer$_j$ as:

$$DV_i = ACT(V_i) - AVT(V_i), \qquad (1)$$

where DV_i is defined as delay time, $ACT(V_i)$ is the actual transmission time of (V_i), $AVT(V_i)$ is the average transmission time of (V_i) and V_i is defined as the i^{th} video stream [55].

Start-up delay is another metric specific to video streaming systems. Startup delay is the amount of time the viewer waits for the video to start up. Once the video starts playing, the average bit rate at which the video was rendered on the viewer's screen is a measure of the richness of the presented content. This metric is somewhat complex since it is a function of how the video was encoded, the network connectivity between the server and the client, and the heuristics for bit-rate switching employed by the player [56]. For streaming applications in best-effort networks, start-up buffering has always been a useful mechanism to deal with the rate variations of streaming sessions. P2P streaming applications additionally have to deal with peer churns, increasing the need for startup buffering and delay [22].

2.1.2 Packet loss

Video streaming content transmitted on the Internet suffers from packet loss, which degrades the video quality [57]. If a video packet cannot be received before its playback time, the reconstructed video quality may be seriously damaged [58]. The packet loss estimation for P2P networks is more complicated than that for traditional client-server structures. Because the video sources are the multiple peers rather than a single server, the packet loss propagates through the inter-peers transmissions. Moreover, peers will usually unexpectedly join and leave the systems [59]. The performance metrics associated with packet loss are frame loss rate and information loss rate [60].

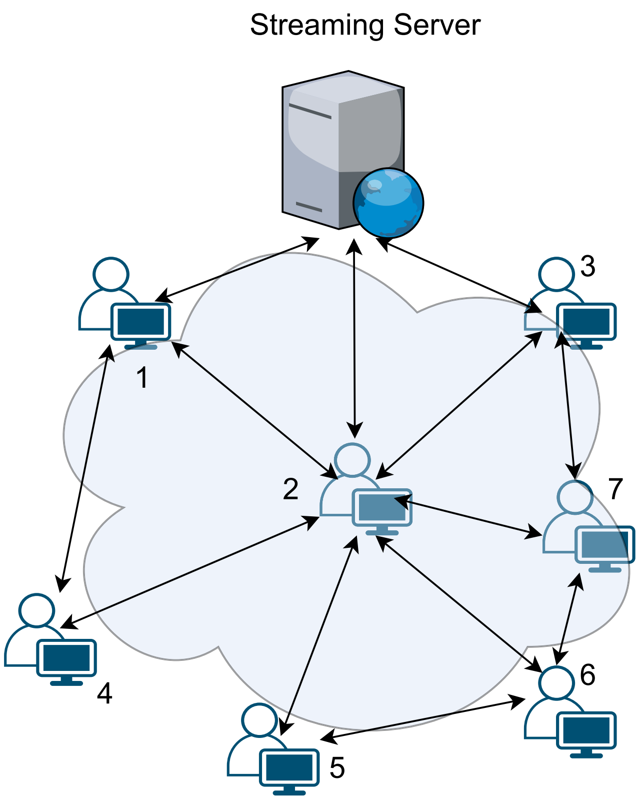


Fig. 4 Mesh based topology [45]

2.1.3 Jitter and playback continuity

Jitter is the maximum delay which is due to variations between two consecutive video streams. It will increase under hectic load patterns. If the jitter is high, play-out process will pause, frustrating the peer [54]. Jitter also determines the acceptable video playback continuity. Jitter must be sufficiently small so as to prevent frequent discontinuities in audio and video playback [61]. Jitter variation is defined as the difference between average packet receiving rate and the average playback rate of the video (that is, variation = average packet receiving rate - average playback rate). The requesting peer or the new joining peer evaluates the quality of each target peer in terms of the video playback time. The score of each target, the requesting peer calculated, reflects the estimated video playback time in the requesting peer if it receives media chunks from that target [54].

2.1.4 Network throughput

Throughput in a network is defined as the number of packets passing through the network per unit time [15]. In peer-to-peer streaming system, throughput is defined as the aggregate of the uplink bandwidth of peers connected to the overlay network [62]. The delivery ratio of the streaming session is measured as the average delivery ratio among all nodes, also representing the throughput of the session [63].

2.1.5 Acceptable QoS parameters for video streaming services

There are several acceptable benchmarks of QoS parameters recommended for video streaming systems. As an example, International Telecommunications Union - Telecommunications Standardization Sector (ITU-T), The 3rd Generation Partnership Project (3GPP) and CISCO have recommended different benchmarks for packet loss and delay for the same service. For video on demand services, ITU-T and CISCO recommend 1% and 3% respectively. Also, packet loss ratio recommendation for live video streaming services are (ITU-T & 3GPP - 3%) and (CISCO - 1%). There is also discrepancy with regards to acceptable values for end-to-end delay. ITU-T and CISCO recommend 10 seconds and 5 seconds respectively for video on demand services. ITU-T and 3GPP recommend 15-400 milliseconds for live video streaming services, while CISCO recommends 150 milliseconds [64]. The maximum jitter for video streaming as recommended by ITU-T is $<$ 50 milliseconds [65]. Lastly, there are recommendations for streaming services of the required data rates, that is, maximum throughput. For instance, ITU-T recommends data rates of 16 to 384 Kbps for video streaming services, while a rate of 384 Kbps is recommended by CISCO. Additionally, performance parameters may vary, depending on the infrastructures and link capacity [64].

3 Related works on P2P streaming topology

The section presents a review of existing research efforts to provide quality of service and experience over a P2P network which serves as motivation for this research.

3.1 Research motivation

This research is motivated by some observations (such as packet loss, end-to-end delay, flash crowd and peer churn) found in the literature. For example, [66] observed that the P2P paradigm has the potential to democratize the streaming world such that everyone may be able to broadcast their media contents. Also, [67] reported a relatively sharp growth in interest among academics and businesses in live streaming using P2P technology leading to P2P digital television (P2PTV) applications (like SopCast [68, 69] and PPLive [22]). Although, [48] explained that the user QoE in current P2P streaming systems are still incomparable to the traditional TV services provided by cable and satellite broadcasting companies. However, [70] emphasized that P2P technology is suitable for distributing multimedia content at low infrastructure cost and collaboration among peers in P2P environments can significantly alleviate server load and enable scalability for a large number of concurrent users. In addition, [19] showed that scalable video coding streaming codec is suitable for streaming video in low bandwidth network. Furthermore, [71] observed that hybrid topology (combination of tree and mesh topologies) can eradicate start-up and end-to-end delays. Lastly, [72, 73] discovered that reconstructing P2P streaming topology, dynamically, can improve the quality of contents. Leveraging on these observations, the implementation of an appropriate overlay topology for video streaming over a P2P network is desirable.

Table 1 gives a summary of existing P2P streaming schemes. The overlay topology applied in each scheme and the main strength of these researches were identified. It was observed that the overlay topology (i.e. the arrangement of peers in the network) plays significant role in the overall performance of these schemes. The tree-based topology models ensure orderliness in the network because peer joins and leaves the network in a fashionable manner as defined by the overlay topology but leading to the

Table 1 Existing research on P2P streaming system

Overlay topology	Authors	Major Strength
Tree-based	[74]	Node failure recovery
	[75]	Excellent fault tolerance and fair load distribution
	[66]	High efficiency and robustness
	[76]	Effective trees construction and low end-to-end delay
	[77]	Greater Throughput
	[25]	Reduction in network traffic
	[78]	Robust streaming in case of peer failure or packet loss
	[79]	Efficient video coding technique and packet-request scheduling
Mesh-based	[80]	High streaming quality and minimal delay
	[81, 82]	Improved global resource utilization and even traffic distribution
	[39]	Achieved low source-to-peer delay and handles peer churn
	[83]	Robust peer selection strategy and minimise peer churn
	[84]	Prevents node failure even in high peer churn and achieved bounded delay
	[85]	Minimised the total rate distortion among receiving peers and provides a better video experience
	[27]	Achieved lower playback delay and better stream continuity
	[86]	Maximise the average network throughput
	[87]	Achieved high streaming rate
	[88]	Achieved low delay and high continuity in the presence of node churns for P2P live streaming. It also achieved high stream continuity
	[89]	Introduced multi-request mechanism in pull-based P2P live streaming system
	[90]	Provides high video quality on nodes by decreasing the amount of video distortion, achieved minimal start-up delay and end-to-end delay
	[91]	Reduces playback lag and startup latency significantly through connection switching while not decreasing playback continuity considerably
Hybrid	[92, 93]	Reduce the initial startup delay and improves the efficiency in data transmission with effective utilization of upload bandwidth .
	[94]	Minimise the startup latencies and transmission delays
	[95]	Minimise frame loss and avoid congestion in the network
	[96]	provide less start-up delay, less transmission delay,less control overheard, more utilization of upload bandwidth
	[97]	improved streaming performance and flash crowd control
	[72, 73]	Prevent node failure and peer starvation
	[98]	Minimal end-to-end delay
	[99]	Better Clustering, end-to-end delay and the peer join latency
	[100]	Treats low-and high-reputed nodes equally, improves QoS, smooth delivery of videos, achieved low end-to-end delay and high frame success ratio
	[101]	Effectively improve network throughput and service performance

major shortcoming of tree-based topology which is the dependability of the children peers on the equivalent parent peers. This implies that a failure at the parent peer connotes a failure to all the attached children's peers which makes it unstable in dynamic environments and large-scale networks. From Table 1, it is observed that most of the tree-based models attempt to design an effective means to recover from node failure or peer starvation. On the other hand, mesh-based system allows peers to join and leave the network without policy enforcement and distribute traffic to all peers evenly, in a situation where any of the parent peers fails, the children peer can directly feed from the next available peer. Mesh-based topology addressed the challenge of tree-based topology with its ability to disseminate chunks directly to the neighbouring peers. However, the mesh-based topology is liable to flash crowd and high churn situation due to unplanned interruptions. The major problem addressed in literatures that is peculiar to mesh-based models is minimising peer churn situation as shown in Table 1. Lastly, the hybrid topology combines the advantages of both tree-based and mesh-based topologies to improve performance of the peers in the network [71]. The existing hybrid-topology based models proved that it can achieve minimal start-up delay, end-to-end delay, control flash crowd and prevent node failure (or peer starvation).

3.2 Existing P2P streaming system analysis

The existing literatures were further analyzed using the following parameters as presented in Table 2.

Node failure/peer starvation: The existing schemes were studied if a preventive or recovery scheme for node failure or peer starvation is applied.

Optimal network load distribution: The existence of network load balancing is also identified in existing literatures.

Delay minimisation: Different types of delay minimisation for existing P2PSS such as; start-up delay, end-to-end delay, peer-join latency, transmission delay, bounded delay, buffering delay, chunk delivery delay and playback continuity were identified.

Throughput maximisation/bandwidth optimisation: In this case, the existing literatures were studied for sufficient throughput or equal distribution (or utilization) of bandwidth.

Video coding techniques: The existing literatures were checked if video coding technique was applied to the system.

Packet scheduling: The utilization of packet scheduling schemes or design of new packet scheduling schemes were observed in the existing literatures.

Packet scheduling: In this situation, the existing models were studied to determine if they would be able to reduce peer churning.

Streaming continuity/quality: The existing literature were checked to determine if either streaming continuity or streaming quality is achieved.

Packet loss ratio: The schemes were also checked for minimal packet loss.

Flash crowd control: The existence of flash crowd control was identified in the existing schemes.

Based on the these parameters, the following are observed:

1) Delay minimisation, streaming quality and throughput maximisation are important performance metrics in P2PSS (from Table 2, it is observed that 95% of existing literatures attempted to achieved these metrics).
2) The tree-based features can effectively prevent or minimise peer churning (Existing models that are designed using tree-based features (either tree-based or hybrid topologies) all achieved minimal peer churn as shown in Table 2).
3) Incorporating video coding techniques or packet scheduling schemes in P2PSS can lead to high streaming quality and effective streaming continuity.
4) The hybrid-based models can successfully control flash crowd.
5) The mesh-based features can effectively prevent or minimise delay.

3.3 Drawbacks from existing schemes

Although, the inability of existing P2P streaming models to effectively prevent high churn, starvation of peers, delay, packet loss and manage transmission of resources among peers is a major concern to researchers. From the literature reviewed in Section 3.2, the limitations of existing schemes are presented below:

1. Waiting time: The existence of longer waiting time before new peers are assigned is observed in existing scheme. For instance, connection switching mechanism was applied in [91] for efficient neighbour peer selection when a new peer joins in a P2P live streaming system. However, if all the active peers have reached the maximum number of neighbours and no peer is available for connection switching, the incoming peers wait until completed peers leaves the network. This waiting period experienced by incoming peers can lead to longer start-up delay, further, this method can not effectively handle multiple arrival and departure of peers (that is flash crowd).
2. Lack of alternate route: The hybrid topology presented in [101] is constructed such that the ordinary peers are

Table 2 Existing P2P streaming system analysis

OT	Authors	ndf/ps	old	dm	tm	vc	mps	mpc	sc	plr	fcc
TB	[74]	✓	✓					✓			
	[75]	✓	✓					✓			
	[66]	✓	✓	PB				✓		✓	
	[76]	✓	✓	E2E	✓	✓		✓	✓	✓	
	[77]	✓	✓		✓			✓			
	[25]		✓	PB, BF & CD				✓	✓	✓	
	[78]	✓	✓		✓	✓	✓	✓	✓	✓	
	[79]		✓			✓	✓		✓		
MB	[80]	✓	✓	TD	✓		✓	✓	✓		
	[81, 82]	✓	✓	S2P&SU	✓		✓	✓	✓		
	[39]		✓	S2P	✓			✓	✓		
	[83]	✓	✓	PL	✓	✓	✓	✓	✓	✓	✓
	[84]		✓	B	✓			✓	✓		
	[85]			✓	✓			✓			
	[27]			PB	✓				✓		
	[86]			✓	MUB				✓		✓
	[87]			✓	✓			✓	✓		
	[88]			✓	✓			✓	✓		
	[89]			✓	✓		✓	✓	✓		
	[90]			SU	✓	✓		✓	✓		
	[91]	✓	✓	SU	✓			✓	✓	✓	
H	[92, 93]		✓	SU	MUB			✓	✓		✓
	[94]		✓	SU & TD	✓			✓	✓		✓
	[95]		✓	✓	✓			✓	✓	Min	✓
	[96]		✓	SU & TD	MUB			✓	✓		✓
	[97]		✓	✓	✓			✓	✓		✓
	[72, 73]	✓	✓	✓	✓			✓	✓		✓
	[98]		✓	E2E	✓			✓	✓		✓
	[99]	✓	✓	E2E & PJ	✓			✓	✓		✓
	[100]	✓	✓	E2E	✓	✓		✓	✓		✓
	[101]	✓	✓	E2E	✓			✓	✓		✓

Legends: OT-overlay topology, ndf-node failure or peer starvation, old- optimal network load distribution or effective peer distribution, dm-delay minimisation, tm-throughput maximisation or bandwidth optimisation, vc –video coding technique, mps-packet scheduling, mpc-minimise peer churning sc- streaming continuity or streaming quality, plr- minimal packet loss ratio, TB-tree-based, MB- mesh-based, H-hybrid, E2E - end-to-end, PJ-peer join latency, S2P-source-to-peer, Min-minimise, Max-maximise, SU- startup,TD-transmission delay, PL- Playout latency PB-playback, B-bounded, BF- buffering delay,CD- chunk delivery delay, MUB- maximise upload bandwidth and FCC-flash crowd control

connected to super peers using overlay tree formation and peers at the same level are connected using mesh overlay formation. However, the provision for alternate route (or approximate peer) is not considered. Hence, if the super node experience network failure before at least one of its child peers complete its download, all peers attached to this super peer automatically experience peer starvation thereby increasing end-to-end delay and packet loss.

3. Network access bottleneck: we observed that most existing P2P streaming systems were tested within high capacity network. The schemes are evaluated based on the assumption that the servers' bandwidth capacity is unlimited [102]. This assumption is not realistic in low-capacity networks where the server's bandwidth capacity is limited.

In an attempt to address the above challenges, this research proposes a more reliable streaming consisting of a new hybrid topology, named ultra-metric spanning tree (UStream) and scheduling strategy scheme which attempts to address the following challenges - high churning, accommodation of unreliable peers, starvation of peers and high packet-loss in low capacity networks.

4 Proposed streaming protocol

The video streaming protocol is conceived as a three layered video streaming protocol - adaptation layer, scheduling layer and topology layer as shown in Fig. 5. Multiple requests are generated from different devices operating across wired to wireless networks. These requests can emanate from several computing machines such as desktops, laptops, palmtops, mobile phones, among others. This work focuses on live video streaming; therefore, requests are assumed to be video frames (chunks). The raw video frames are admitted directly by the adaptation layer which in turn encodes frames using standard video codec to reduce the frame size but retains quality. The adaptation layer adopts the existing scalable video coding (SVC) smoothing function [19]. The second layer, the scheduling layer, accepts the encoded video frames from the adaptation layer and distributes using traffic scheduling scheme to avoid congestion [103].

The scheduling scheme has the capacity to adapt to different network conditions such as slow and fast speed networks. The topology layer represents the overlay structure of P2P streaming system. The scheduled video frames are transferred directly to the topology layer. The topology layer monitors the arrival and departure of each peer within the network and ensures that the requested video chunks are disseminated within the set time frame. Furthermore, the layer guarantees successful transmission of video chunks, reduction in packet loss rate and avoidance of interruption from unsolicited peers. The output video chunks from the topology layer are delivered to the required destination. At the receiving end, the request moves from the topology layer to the scheduling layer and then to the adaptation layer which decodes the frames and sends frames to the play-out buffer, where they are received by the desired user's device. To further explain this protocol, the streaming protocol is mapped to transmission control protocol and Internet protocol (TCP/IP) as displayed in Fig. 5. Raw videos are sent directly to the adaptation layer which ensures that video frames are encoded/decoded. The adaptation layer of the proposed streaming protocol is located at the application layer of TCP/IP model. The encoded video frames now move from the application layer to the transport layer. In addition to the functions perform at the transport layer, this protocol also schedule frames at the transport layer and the transport layer of the streaming protocol is called scheduling layer. The schedule frames are passed on to the Internet layer which is called the topology layer in the streaming protocol. Specifically, this research designed a new overlay structure for the topology layer

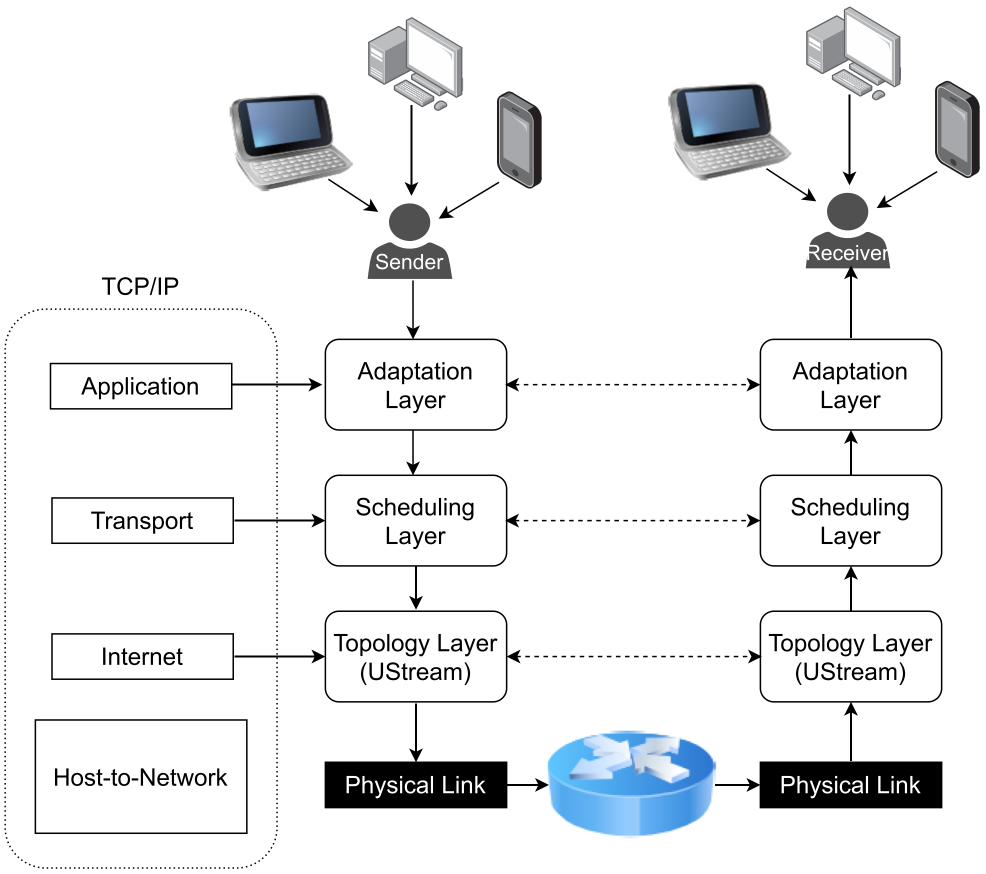


Fig. 5 Proposed streaming protocol and TCP/IP model mapping (adapted from [36])

that is domiciled at the Internet layer of the TCP/IP model. Packets move from the Internet layer to host-to-network layer using physical links and then to the receiver side which also goes through similar process like the sender side.

4.1 Scheduling layer

The steady traffic scheduler (STS) is a congestion control scheme that is specifically designed for large delay networks. It is an enhanced version of the optimised video scheduling model (OVSM) [103]. STS is located before the Internet layer. It schedules encoded video frames from the application layer before it gets into the Internet. This scheme implements the features of both weighted fair queuing and leaky bucket traffic sharper. It accepts unregulated video frames as input, groups the frames into various buffers based on the weight of the frame and selects frames using time slice. These frames are regulated within the leaky bucket to ensure constant flow of frames which serves as input into the network. To achieve this, STS was divided into three main modules as depicted in Fig. 6 - input module, frame scheduler module and the traffic shaper module. The input module is the first phase of the system, accepts incoming group of pictures(GOP) from the user. A classifier is embedded within the input module that groups frames within the GOP using weighted fair queuing technique which classified frames based on the type. For the purpose of this work, three basic frames were considered: the I frame, the P frame and the B frame. The I frames are given the highest priority because they contain the most important frames, followed by the P frames and the B frames. Therefore, any time the I frame is encountered, the classifier ensures that it flows into the next phase without any delay and the rate at which it flows is higher compared to other frames. The P frames are given the next level of priority followed by the B frames.

The second module is the frame scheduler. At this phase, frames are selected from different buffers based on the weight attached to them by the classifier. Time controller is the major component in this phase that ensures frames at the lower priority don’t wait unnecessarily: therefore, the time controller have a time slices attached to each frame to prevent starvation. So, the time controller picks frames randomly from the weighted queues and places them in the last module . The last module is the traffic sharper. This phase is designed based on leaky bucket technique. It accepts unregulated video frames from the frame scheduler phase as input. The unregulated video frames are processed

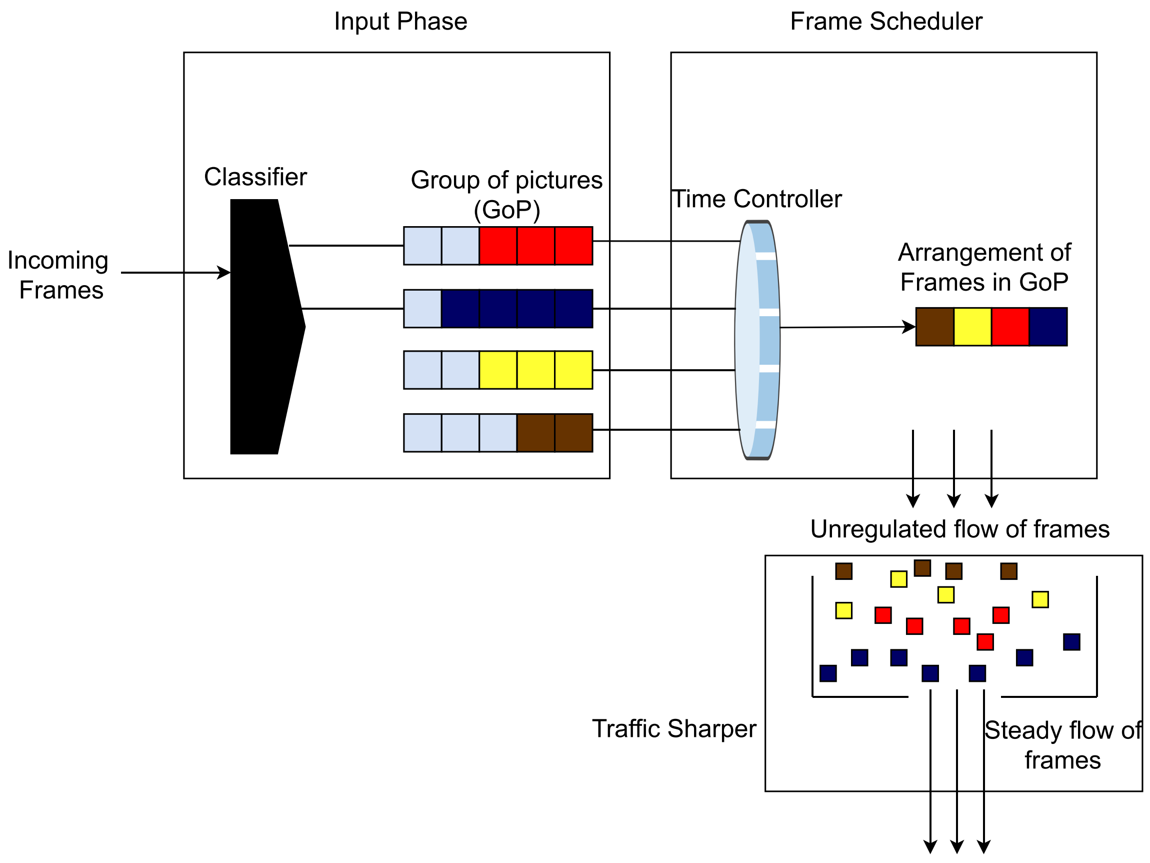


Fig. 6 The steady traffic scheduler

within the traffic sharper to give steady flow of frames; therefore, the output from the traffic sharper module are steady video frames which go directly into the network. The STS is a congestion control mechanism that is suitable for large delay networks. The steady flow of video traffic avoids congestion within the network to the barest minimum, at the same time, allowing relevant frames within a GoP to flow without any delay.

4.2 Topology layer

The overlay topology used in this work is named UStream as depicted in Fig. 7. UStream consists of the routing server, rooted peers, parent peers and children peers. The routing server manages the arrival and departure of peers in the network. When a new peer joins the network, the routing server ensures that incoming peers are not starved or delayed unnecessarily without any activity. Also, when a peer disconnects or leaves the network, the routing server ensures that all peers associated or connected directly or indirectly are relocated to the next available active peer.

The root peers are origin initial peers that transmit or channel encoded video frames to all other peers. The root peers are the first layered peers that transport video frames to all the peers at the next layer. The parent peers at the second layer of the UStream topology receives video frames or data chunks directly from the root peers which in return transmit data chunks to the children peers. The children peers can also connect directly to the root peer in situations where the corresponding parent peers failed. The UStream adopts the features of ultra-metric tree.

UStream was constructed using discrete topology formula ($P(\mathcal{N}) = 2^{2^{\mathcal{N}-1}}$), where $\mathcal{N}$ is the set of natural number. The topology is a three-level layered topology. At the first layer, the initial peers known as the root peers are represented by n_0. This n_0 is a great determinant in the number of peers that will be represented at the second layer known as parent peers. Assume that $n_0 = 2$, then the parent peer $n_1 = 2^2 = 4$. Furthermore, n_1 also determines the number of peers at the third layer called children peers (n_2), that is, ($n_2 = 2^4 = 16$). If we assume an initial value $n_0 = 10$, then $n_1 = 2^{10}$ and $n_2 = 2^{2^{10}}$ = approximate values in octillion. This implies that there are more than octillion users requesting for video chunks at the same time. This might not be realistic, because of insufficient system memory. The limit for which the system can handle can be obtained using a combinatorial analysis such that for n peers and r pairs of two elements the constraint is given by (${}^{n}C_r$), where $r = n_0 - 2$ and $n_0 \geq 3$. For example, suppose $n_0 = 50$, then ${}^{n}C_r$ = ${}^{50}C_{48}$ = 1225.

4.2.1 UStream formalization

Assumption i: peers on the same level have the same attributes (same bandwidth).

Assumption ii: When a peer is active, it is assumed that it is 1 and 0 when it is not active.

Assumption iii: Let a and b represent peers and d the distance function. Peers are equal when they are inactive and their distance is zero, that is, $d(a, b) = 0$.

Assumption iv: Let $A = A_i$, $B = B_j$ be peers on different levels L_1 and L_2 respectively. The distance

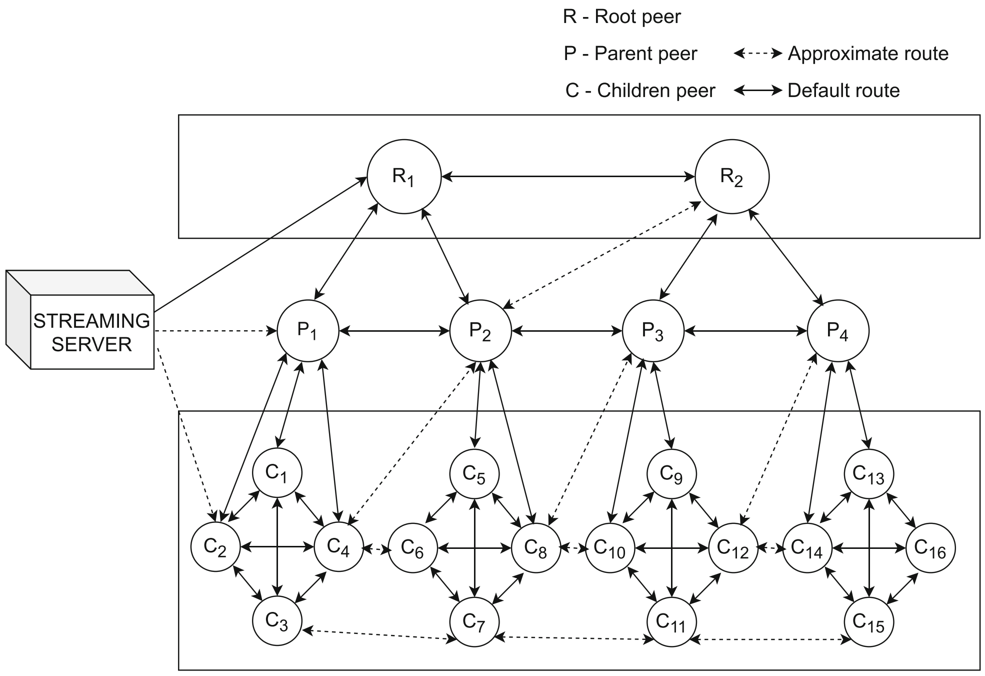


Fig. 7 Architecture of UStream

between peers at different levels are always positive, that is, $d(A_i, B_j) > 0$.

Assumption v: The arrival/departure time of given peers on the network are not necessarily the same.

Lemma 1 *Let $A = A_i$, $B = B_j$ be peers on different levels L_1 and L_2 respectively. The distance d between two peers on L_1 and L_2 are always equal. Thus, $d(A_i, B_j) = d(B_j, A_i), \forall i, j \in N$*

Remarks: Consequent on Assumption (i), peers of the same attributes are designated to a particular level. Thus, the distance between any two peers on different levels are equal.

Theorem 1 *If parent peers P_i and P_j are sub-trees of active root peers R_1 and R_2 respectively, let a and b be the chunk request from P_i and P_j respectively. Given that t_a and t_b are the delivery times to the parent peers, then $t_a = t_b$.*

Proof From the UStream topology formula ($P(\mathcal{N}) = 2^{2^{\mathcal{N}-1}}$), the root peers have two peers R_1, R_2 (since $\mathcal{N} = 1$).

Similarly, the parent peers have four peers namely $\{P_1, P_2\}$ and $\{P_3, P_4\}$ (since $\mathcal{N} = 2$).

Let $\{P_1, P_2\} = P_i$ and $\{P_3, P_4\} = P_j$.

suppose the mapping f and g be such that:

$f : R_1 \mapsto P_i$ and

$g : R_2 \mapsto P_j$

and defined by

$f(a) = t_a \Rightarrow$ time to parent peers P_i,

$g(b) = t_b \Rightarrow$ time to parent peers P_j.

From Lemma 1

$d(R_1, P_i) = d(R_2, P_j)$

Since the bandwidth of R_1 and R_2 are equal (same level), this implies same speed. Thus,

Speed = $\frac{d(R_i,P_j)}{t}$.

Hence, $\frac{d(R_1,P_i)}{t_a} = \frac{d(R_2,P_j)}{t_b}$

therefore $t_a = t_b$. □

Theorem 2 *If children peers C_i, C_j, C_k and C_l are sub-trees of active parent peers P_1, P_2, P_3 and P_4 respectively. Let c be the chunk requested by C_i, d the chunk requested by C_j, e the chunk requested by C_k, and f the chunk requested by C_l from P_1, P_2, P_3 and P_4 respectively. Given that t_c, t_d, t_e and t_f are the delivery times to the children peers, then $t_c = t_d = t_e = t_f$.*

Proof The proof follows from Theorem 1. □

Proposition 1 *If root peer R_1 is active and the path R_1 to P_i, R_1 to P_j are default route, then R_1 determines the stability of P_i and P_j at a time t.*

Remarks The high level peers determine the behaviours of the low level peers and the presence of the high level peers result in the stability of the system, while the absence may lead to instability since the system will have to adjust itself.

Proposition 2 *If $x = \{x_n, x_m, ...x_k\}$, where $n, m, ..k$ are natural number $\leq |N|$, N = set of peers per level in the UStream topology. therefore, there exists a linear connection such that $x_n \rightarrow x_m ... \rightarrow x_k$ at the peer level.*

Remarks Peers on the same level are interconnected except if there is failure among any on the peer level.

Case 1 : It is obvious from the UStream topology provided that all peers are active at the same level ($x_n \rightarrow x_m ... \rightarrow x_k$).

Case 2 : Assume that at least a number of the peers is inactive, then, the existence of alternate route becomes inevitable. Thus, at the point of discontinuity the connectivity is now virtual($x_n \rightarrow x_m ... \dashrightarrow x_k$).

Lemma 2 *Every approximate peer may not function as a default peer.*

Conditions:

(i) *If the approximate peer is at the same level with the default peer, there is a possibility that it may deliver like the default peer.*

(ii) *If the approximate peer is at the level different from the default peer, then the quality of chunks and the delivery time are different.*

Proof for Case 1 From Theorem 1, assume that R_y becomes inactive and P_j depends on R_x for requested chunk, thus becoming an approximate peer.

Then the mapping f and g becomes:

$f : R_x \mapsto P_i$ and

$g : R_x \mapsto P_j$.

If a and b are chunk request by P_i and P_j from R_x, then

$f(a) = t_a \Rightarrow$ time to parent peers P_i,

$g(b) = t_b \Rightarrow$ time to parent peers P_j.

From Lemma 1

$d(R_x, P_i) = d(R_x, P_j)$

Since the bandwidth of P_i and P_j are equal (same level), this implies same speed. Thus,

Speed = $\frac{d(R_i,P_j)}{t}$.

Hence, $\frac{d(R_x,P_i)}{t_a} = \frac{d(R_x,P_j)}{t_b}$

therefore $t_a = t_b$. □

Remark Although, P_i and P_j receive chunks from R_x at the same time, R_x may experience a higher delivery time because of its dual functions.

Proof for Case 2 Assume that R_y becomes inactive in Theorem 1 and P_j depends on P_i for relevant chunks, that is, becomes an approximate peer.

such that, the mapping (time) f and g becomes:
$f : R_x \mapsto P_i$ and
$g : P_i \mapsto P_j$. defined by
$f(a) = t_a \Rightarrow$ time to parent peers P_i,
$g(b) = t_b \Rightarrow$ time to parent peers P_j.
From Assumption (iv),
$d(R_x, P_i) > 0$ and $d(P_i, P_j) = 0$
Hence, $t_a \neq t_b$. □

5 UStream and STS algorithms

This section presents the STS algorithm and UStream algorithms; ultra-metric tree algorithm, spanning tree algorithm, behaviour of peers and approximate peers. The list of terms for the algorithms is given in Table 3. In addition, the time complexity of the algorithms is also presented in this section.

5.1 STS Algorithm

The process involved in the scheduling layer described in Section 4.1 is represented in the STS algorithm presented in Algorithm 1. The following physically reasonable assumptions were adopted in Algorithm 1:

(i) Encoded video frames from the adaptation layer flow into the input phase at 25 frames per seconds ($\lambda_i = 25$), utilizing the scalable video coding standard;

(ii) The frame scheduler module uses time controller which sends frames at a certain threshold ($\alpha_t = 10$ nanoseconds) and

(iii) The size of the leaky bucket was assumed to be much larger than frame scheduler module buffers (Fbs).

The input phase frames, scheduler frames, leaky bucket frames, weight attached to GoP and leaky bucket buffer are initially set to zero. The STS algorithm accept video frames as input and frames are classified based on frame types which are stored to their respective buffer as shown in Lines 4 -13 of Algorithm 1. Further, the time controller releases frames based on priority tag as given in Lines 14 -22. When frames arrive into the leaky bucket buffer, the algorithm releases frames at constant rate based on computed size of the leaky bucket hole.

Table 3 list of terms for algorithms

Symbols	Meaning
λ_i	frames at the input phase
λ_j	frames at the frame scheduler
λ_k	frames at the leaky bucket
ω_i	weight attached to group of frames
β	leaky bucket (space)
ζ	size of hole of the leaky bucket
n	total number of frames in the leaky bucket
r_x	the first layer set of peers classified as root peers
p_x	second layer set of peers named parent peers
c_x	set of peers at the third layer- children peers
d	distance between peers
$G(P, E)$	graph G is a pair $G = (P, E)$
P	set of peers
E	set of weight along the peers
α	requests from peers
a	approximate peers

Algorithm 1 : STS algorithm.

Initialization : $\lambda_i, \lambda_j, \lambda_k, \omega_i, \beta$

```
1:  $\lambda_i \leftarrow$ video frames
2:  Frame type = enum {Iframe, Bframe, Pframe}
3:  buffer: Ibuffer() , Pbuffer(), Bbuffer()
4:  Begin
5:      let $\lambda_i = \{\lambda_1, \lambda_2, ...\lambda_n\}, (n \in Z^+)$
6:      Begin{Classify frames}
7:          Case(Iframe)
8:          $\lambda_i$ = Iframe
9:          store Ibuffer $\leftarrow \lambda_i$
10:         Case(Pframe)
11:         $\lambda_i$ = Pframe
12:         store Pbuffer $\leftarrow \lambda_i$
13:         Case(Bframe)
14:         $\lambda_i$ = Bframe
15:         store Bbuffer $\leftarrow \lambda_i$
16:     End{Classify frames}
17:     Begin{Release frames}
18:         for $i$ : 1 to TotNum,
          TotNum $\leftarrow$ find total number of frames
19:         if $\lambda_i \in$ Iframe(i)
20:         release $\lambda_i$ at $\alpha(t)$
21:         else if $\lambda_i \in$ Pframe(i)
22:         release $\lambda_i$ at $\alpha(t) + 1$
23:         else $\lambda_i \in$ Bframe(i)
24:         release $\lambda_i$ at $\alpha(t) + 2$
25:         end if
26:         repeat until $\lambda_i = 0$
27:         end for
28:     End{Release frames}
29:     arrival of $\lambda_j$ to $\beta$
30:     compute $\zeta$,( $\zeta = \sum_{\iota=0}^{n} \omega i/n$ )
31:     set $\lambda_k \leqslant \zeta$
32:     repeat until $\lambda_k = 0$
33: End
```

5.2 UStream algorithms

It is assumed that peers are organized in a hybrid topology for all the algorithms in UStream. The ultra-metric tree concept is represented in Algorithm 2. The ultra-metric tree algorithm remains valid as long as the root, parent and children peers exist in the UStream topology. The initial state of all peers in the UStream topology is set to zero and the state of these peers is activated by chunk request. In addition, the initial state of the root peers is also set to zero which is activated when a streaming session begins. Whenever, the state of the root peers is active (that is, state ($r_x = 1$)) and there is chunk request from the parent state ($p_x = 1$)). Therefore, the distance function from the root peers to its corresponding parent peers as discussed in Section 4.2 is triggered as indicated in Lines 7-11 of Algorithm 2. Similarly, if the route from the parent to children peers is activated based on chunk request (that is, state ($p_x = 1$) and state ($c_x = 1$)) then, the distance function is as well activated for the parent peers to children peers as presented in Lines 15-19 of Algorithm 2.

Algorithm 2 : Ultra-metric tree algorithm.

```
1: Begin
2:     for all $(r_x, p_x, c_x) \in E$ do
3:     state $((r_x, p_x, c_x)) = 0$
4:     end for
5:     for all $(r_x) \in P$ do
6:     state $(r_x) = 0$
7:     end for
8:     $r_x = r_0$
9:     state $(r_x, p_x) = 1$
10:    while $\exists r_x, p_x \in P$ do
11:    $d(r_x, p_x) = d(p_x, r_x)$
12:    end while
13:    for all $(p_x) \in P$ do
14:    state $(p_x) = 0$
15:    end for
16:    $p_x = p_0$
17:    state $(p_x, c_x) = 1$
18:    while $\exists p_x \in P$ do
19:    $d(p_x, c_x) = d(c_x, p_x)$
20:    end while
21: End
```

The feature of the spanning tree is represented in Algorithm 3. Similar to Algorithm 2, the initial state of all peers in the UStream topology is set to zero to represent inactivity before the streaming session. The distribution of video chunks for the peers at the root level based on peers connectivity is activated, whenever the state of the root peers is set to one (that is, state ($r_x = 1$)) as represented in Lines 8-11 of Algorithm 3. Further, the connection of peers at the parent level is given in Lines 16-19 and Lines 24 -27 activates the spanning tree function for the children peers.

Algorithm 3 :Spanning tree algorithm.

```
1: Begin
2:     for all $(r_x, p_x, c_x) \in E$ do
3:     state $((r_x, p_x, c_x)) = 0$
4:     end for
5:     for all $(r_x) \in P$ do
6:     state $(r_x) = 0$
7:     end for
8:     $r_x = r_0$
9:     state $r_x = 1$
10:    while $\exists r_x \in P$ do
11:    $r_{x1} \mapsto r_{x2} \mapsto r_{xn}$
12:    end while
13:    for all $(p_x) \in P$ do
14:    state $(p_x) = 0$
15:    end for
16:    $p_x = p_0$
17:    state $p_x = 1$
18:    while $\exists p_x \in P$ do
19:    $p_{x1} \mapsto p_{x2} \mapsto p_{xn}$
20:    end while
21:    for all $(c_x) \in P$ do
22:    state $(c_x) = 0$
23:    end for
24:    $c_x = c_0$
25:    state $c_x = 1$
26:    while $\exists c_x \in P$ do
27:    $c_{x1} \mapsto c_{x2} \mapsto c_{xn}$
28:    end while
29: End
```

The approximate peers function is represented in Algorithm 4. The initial state of the root peers is set to zero, however, if the root peer is active and at some point during transmission the state of the root peers is reset to zero due to node failure, then, the approximate root peer is activated. As shown in Lines 9-19 of Algorithm 4, there are two alternatives to creating approximate root peer. First, if all the root peers are disconnected from the streaming session (that is $p_x > r_x$) , therefore, the parent peer with highest bandwidth and current complete chunks will become the approximate root peer. The second option is activated if at least one of the neighbouring root peers is active and holding the requested chunks (that is, $ar_x > ap_x$). The algorithm also applies the same rule for creating approximate parent peer as shown in Lines 21-31 of Algorithm 4.

Algorithm 4 :Approximate peers.

```
1:  Begin
2:      for all $(r_x, p_x, c_x) \in E$ do
3:      state $(r_x, p_x, c_x) = 0$
4:      end for
5:      for all $(r_x) \in P$ do
6:      state $(r_x) = 0$
7:      end for
8:      $r_x = r_0$
9:      state $r_x = 1$
10:     while $\exists r_x \in P$ , state $(r_x) \equiv 0$ do
11:     let $ar_x = -r_x$ and $ap_x = -p_x$
12:     if $ar_x - ap_x > 0$
13:     $\Rightarrow -r_x - (-p_x) > 0$
14:     $\Rightarrow -r_x + p_x > 0$
15:     $\Rightarrow p_x > r_x$
16:     also, assume that $-p_x < -r_x$
17:     $\Rightarrow -r_x > -p_x$
18:     $\Rightarrow ar_x > ap_x \Leftrightarrow r_x \neq 0$
19:     end if
20:     end do
21:     $p_x = p_0$
22:     state $p_x = 1$
23:     while $\exists p_x \in P$ , state $(p_x) \equiv 0$ do
24:     let $ap_x = -p_x$ and $ac_x = -c_x$
25:     if $ap_x - ac_x > 0$
26:     $\Rightarrow -p_x - -c_x > 0$
27:     $\Rightarrow -p_x + c_x > 0$
28:     $\Rightarrow c_x > p_x$
29:     also, assume that $-c_x < -p_x$
30:     $\Rightarrow -p_x > -c_x$
31:     $\Rightarrow ap_x > ac_x \Leftrightarrow p_x \neq 0$
32:     end if
33:     end do
34: End
```

Lastly, the behaviour of peers in the UStream topology is implemented in Algorithm 5. As given in lines 9-15 of the algorithm, the activities of the parent peers are activated based on the states of the root peers. Similarly, the activities of children peers are determined by the states of its parents peers as represented in lines 20-27.

5.3 Complexity of the proposed algorithm

The complexity of the STS and UStream algorithms are computed using the time complexity method. The time complexity of the STS algorithm is calculated as $O(n)$, where n is the total number of frames. In addition, the time complexity is also computed for all the UStream algorithms. The time complexity of the ultra-metric tree algorithm is given as $O(n^3) = n^3 + m + k$, where n is assumed as the maximum numbers of root (n), parent (m) and children (k) peers. The spanning tree algorithm is computed as $O(n^3) = n^3 + n + n + m + m + k + k$, where n is assumed as the maximum number of root (n), parent (m) and children (k) peers. Furthermore, the approximate time complexity is calculated as $O(n^3) = n^3 + n + n + m$, where n is assumed as the maximum numbers of root (n) and parent (m) peers. The behaviour of peers algorithm is computed as $O(n^3) = n^3 + n + n + m + m$, where n is assumed as the maximum numbers of root (n) and parent (m) peers.

Algorithm 5 :Behaviour of peers.

```
1:  Begin
2:      for all $(r_x, p_x, c_x) \in E$ do
3:      state $((r_x, p_x, c_x)) = 0$
4:      end for
5:      for all $(r_x) \in P$ do
6:      state $(r_x) = 0$
7:      end for
8:      $r_x = r_0$
9:      state $r_x = 1$
10:     while $\exists r_x \in P$ , state $(r_x) \equiv 0$ do
11:     $r_x - p_x > 0$ when $r_x \neq 0$
12:     suppose $r_x - p_x \geq 0$ and $r_x = 0$
13:     then $r_x - p_x = 0$, $p_x = 0$
14:     also, if $r_x - p_x > 0$
15:     then $0 > p_x$ and $p_x < 0$
          if $r_x$ fails, $p_x$ becomes the approximate root peer
16:     end if
17:     for all $(p_x) \in P$ do
18:     state $(p_x) = 0$
19:     end for
20:     $p_x = p_0$
21:     state $p_x = 1$
22:     while $\exists p_x \in P$ , state $(p_x) \equiv 0$ do
23:     $p_x - c_x > 0$ when $p_x \neq 0$
24:     suppose $p_x - c_x \geq 0$ and $p_x = 0$
25:     then $p_x - c_x = 0$, $c_x = 0$
26:     also, if $p_x - c_x > 0$
27:     then $0 > c_x$ and $c_x < 0$
          if $p_x$ fails, $c_x$ becomes the approximate parent peer
28:     end if
29: End
```

6 Performance evaluations

In this section, the experiment setup for simulation and the results obtained from the event-driven simulation are discussed. A scalable video coding real-time video trace file (that is H.265 /HEVC) obtained from [104] was used as test data for all the simulation scenarios; the H.265 file consists of I frames, P frames and B_{avg} frames. For closer observation of the behaviour pattern of the existing and proposed streaming protocols, all the simulation scenarios were subjected to a wireless simulation environment.

6.1 Experimental setup

The performance of the proposed streaming protocol is tested within a packet-level simulation environment, in particular, the STS and UStream algorithm presented in Section 5 is implemented within the TCP/IP suite in network simulator 3 (NS3) using NS3.28.1 version. The proposed scheme is benchmarked against the neighbour selection algorithm [91] and node selection method [101]. For simplicity, the models presented in [91] and [101] are abbreviated as “Kim-NSA” and “Rongfe-NSM” respectively. The graphical representation of the proposed streaming protocol in NS3 is depicted in Fig. 8.

A new module named “ustream” is created at the source(src) in NS-3.28.1. The ustream module inherits some features from two parent modules - application and mesh modules. The ustream was designed using the standard format in NS-3, it contains model, helper and wscript. The wscript is written in python programming language and it binds all the C++ programs in the module together. To represent the adaptation layer of the streaming protocol in NS-3, the H.265 trace file is converted to acceptable NS-3 video trace format using awk script. The STS algorithm is also converted to C++ codes and implemented as steady-traffic.cc and steady-traffic.h in ustream module. In addition, the STS algorithm accept the H.265 video trace file as input. Furthermore, the ustream algorithms are written in C++ programming language which are saved in the model; the spanning tree, ultra-metric tree, behaviour of peers and approximate peers are implemented in “ustream.cc” and “ustream.h” supported by “ustream-stack.cc” and “ustream-stack.h”. The helper contains the “ustream-helper.cc”, “ustream-helper.h”, “sts-helper.cc” and “sts-helper.h”, which make it easier to create an instance of the model. To test the effectiveness of the ustream module, a wireless network simulation scenario (NS3/multirate.cc) is adopted. The summary of the simulation parameters is given in Table 4.

In the simulation scenario, a 15 by 15 grid topology is created to make a total of 225 peers in the network, multiple simultaneous flow, multi-hop ad-hoc routing and mobility are also embedded in the scenario. The grid topology is divided into four quadrants. To depict P2P network, the source and destination nodes are randomly selected such that node maybe source for multiple destinations and/or destination for multiple source. The sender node (representing root peer) setup flow to each of its neighbours (parent peers or children peers) in its quadrants and all flows are exponentially distributed. The source is equipped with H.265 video trace file and scheduling mechanism is performed. The ustream overlay topology is installed on all the nodes by creating an instance of “UstreamHelper”. The node distance in the simulation scenario varies between 15 meters and 30 meters and the total simulation period is 200 seconds. The simulation scenario runs severally while varying the peers bandwidth using three categories- average speed (5 Mbps - 25 Mbps), slow speed (1 Mbps - 4.5

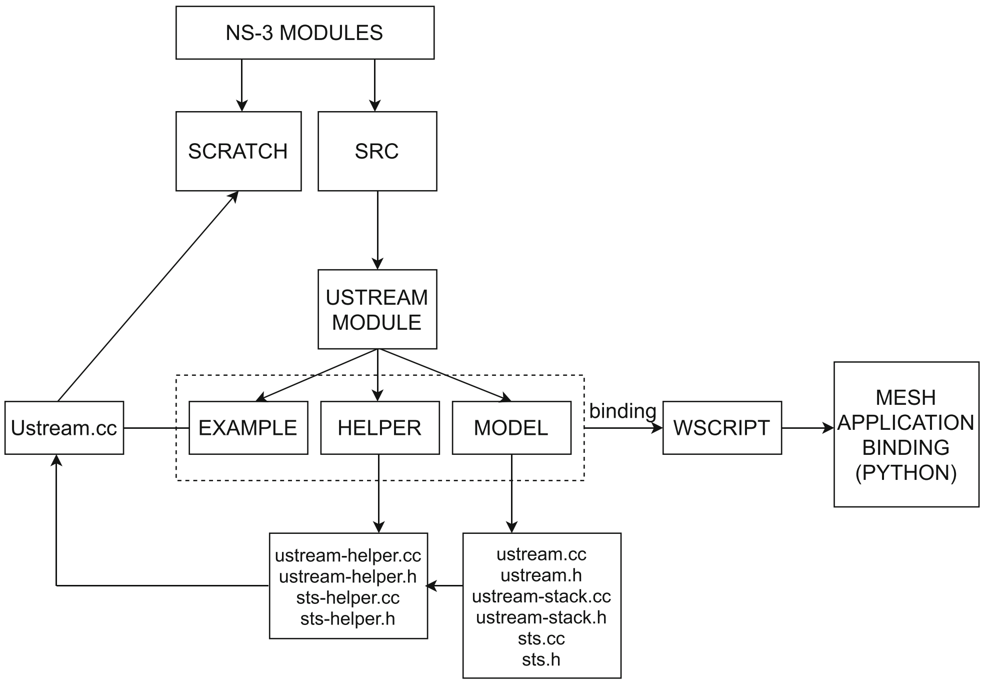

NS-3 MODULES
SCRATCH
SRC
USTREAM MODULE
Ustream.cc
EXAMPLE
HELPER
MODEL
binding
WSCRIPT
MESH APPLICATION BINDING (PYTHON)
ustream-helper.cc ustream-helper.h sts-helper.cc sts-helper.h
ustream.cc ustream.h ustream-stack.cc ustream-stack.h sts.cc sts.h

Fig. 8 Streaming Protocol in NS3

Table 4 Simulation parameters

Variable	Value
Node placement model	random distribution
Node distance	15m - 30m
Total number of nodes	225
Bandwidth Assignment	0.1 Mbps - 25 Mbps
Backbone Network	100 Mbps
Video file	Speed - 1920 × 1080.yuv
Flows	exponentially distributed
Codec	H.265/HEVC
Total simulation time	200s
Transmission delay	2 ms - 5ms

Mbps) and extreme slow speed (0.1 Mbps -0.9 Mbps). The existing schemes - “Kim-NSA” and “Rongfe-NSM” are also tested using this simulation scenario. The output trace file is generated using flow monitor module in NS3 and analyzed for mean delay, mean jitter, packet loss ratio, throughput, playback continuity and start-up delay.

6.2 Simulation results -mean delay

The graph of the mean delay for average speed network category is presented in Figs. 9–10. The mean delay measures the average end-to-end delay for the proposed scheme and existing schemes. Figure 9 compares the mean delay of the three streaming models without STS algorithm. Instability is observed as the traffic flow ramps up to 20. Whereas, UStream presents a start-up delay of 0.003 second which later stabilises at 0.0004 second as the traffic flow increases to 200. On the other hand, Kim-NSA and ”Rongfe-NSM produced a start-up delay of 0.0043 second but later stabilises at 0.008 second and 0.007 second respectively as traffic flow increases from 0 to 200. The instability observed in the three schemes is indicative of multiple requests from several peers at the beginning of the live streaming session but UStream is able to reduce the start-up delay by 30% as well as a 43% and 50% reduction in mean delay in comparison to Rongfe-NSM and Kim-NSA respectively. The results of mean delay for an average speed network as STS algorithm is tested alongside with UStream and in comparison to the two existing schemes is shown in Fig. 10. It is discovered that the UStream lowered the start-up delay to 0.00145 second from 0.00155 second and 0.003 second in Rongfe-NSM and Kim-NSA respectively. Again at stability, UStream lowered mean delay to 0.0003 second from 0.0006 second for Rongfe-NSM and 0.00075 for Kim-NSA. In this scenario, UStream outperforms Rongfe-NSM by reducing start-up delay by 6.5% In the overall, UStream outperforms Rongfe-NSM by 50% and Kim-NSA by 60%. The implication of this result is that UStream combined with STS algorithm can effectively handle flash crowd and peer churn situations in P2P streaming systems.

For slow-speed network category, the mean delay for the three streaming schemes are compared as depicted in Fig. 11. The results showed UStream accomplished a minimal start-up delay of ≈ 0.0088 second as against 0.011 and 0.0095 for Rongfe-NSM and Kim-NSA respectively. This indicates that a maximal of 74% reduction in start-up delay is achieved by UStream. Furthermore, it is observed that Rongfe-NSM and UStream produced the same mean delay of 0.0018 second while the mean delay is 0.0025 second for Kim-NSA. This shows that UStream exhibits the same performance with Rongfe-NSM but, it outperformed Kim-NSA by 28%. The mean delay results of the three streaming protocols with STS algorithm for slow speed

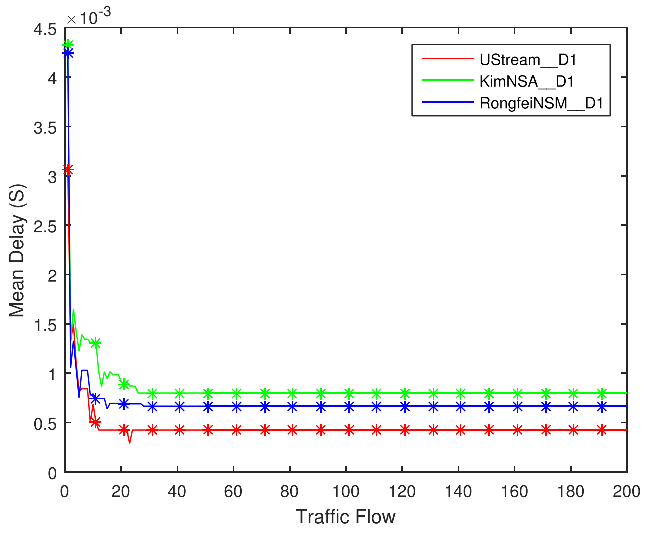


Fig. 9 Average speed - mean delay

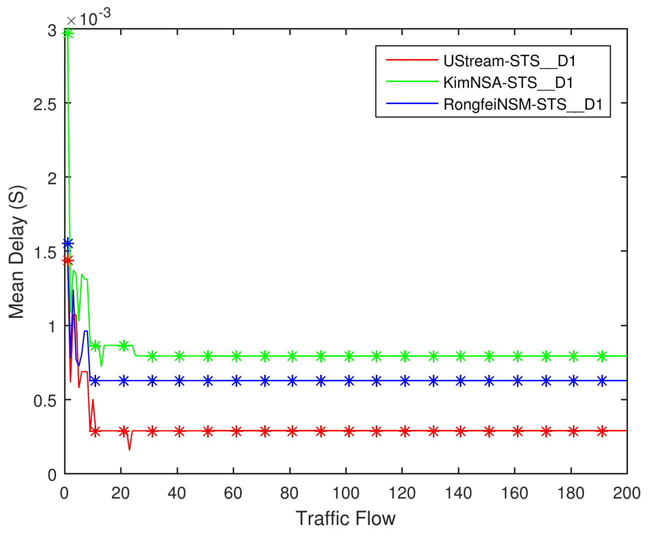


Fig. 10 Average speed-mean delay(with STS)

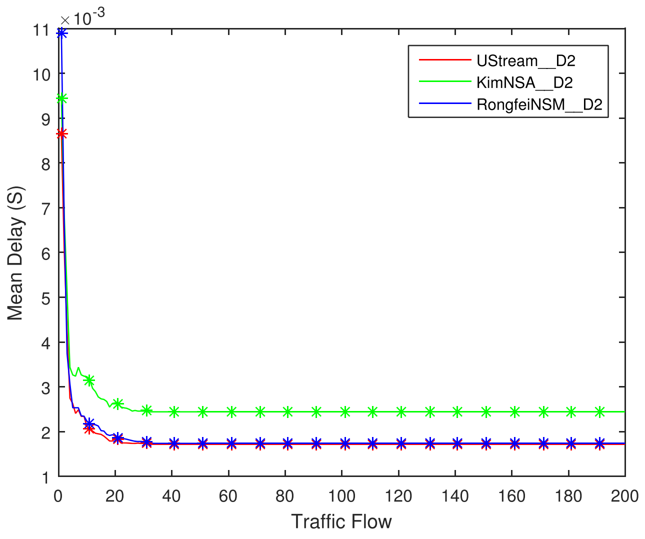


Fig. 11 Slow speed - mean delay

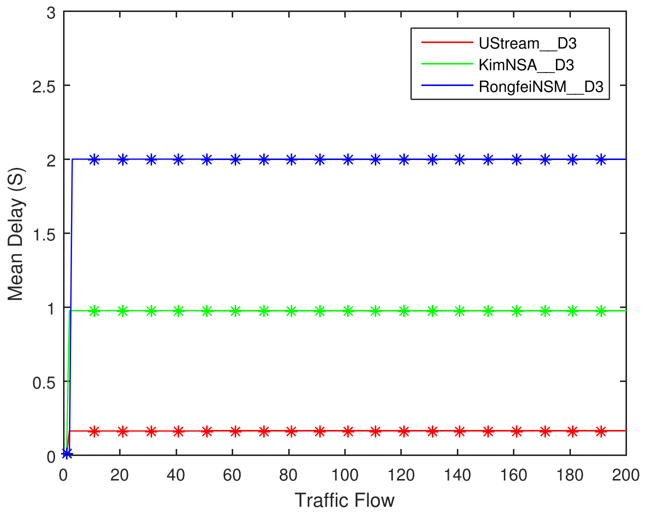


Fig. 13 Extreme slow speed - mean delay

network are evaluated in Fig. 12. The graph shows that UStream outperforms the Rongfe-NSM and Kim-NSA with a start-up delay of 0.0072 second as against 0.0101 second and 0.009 second. This implies that UStream achieved a decrease in start-up delay by 28.7% and 20%.

In extreme slow speed environment, mean delay results for the three streaming protocols as presented in Fig. 13. The graphs are quite different from average and slow speed mean delay results. For all the tested streaming protocols, it generated high delay. The UStream maintained a steady mean delay $\approx$ 0.2 second till the end of the simulation period while Rongfe-NSM and Kim-NSA produced increased mean delay of 1 second and 2 seconds respectively. This indicates that UStream decreased delay with minimum of 80% and maximum of 90%. However, this high delay generated for this scenario is significantly minimised when all the streaming protocols were combined with STS algorithms as depicted in Fig. 14. The result revealed that UStream and Rongfe-NSM produce similar mean delay of $\approx$ 0.018 second as against $\approx$ 0.16 for Kim-NSA.

6.3 Simulation results - mean jitter

The mean jitter results generated for all the simulation scenarios are presented in this sub-section. Figure 15 compares the mean jitter of the three streaming protocols for average speed. The graph shows that UStream outperforms Rongfe-NSM and Kim-NSA with maximum mean jitter of 0.0002 second as against 0.000026 and 0.0007 seconds. Furthermore, it is observed from all the three streaming

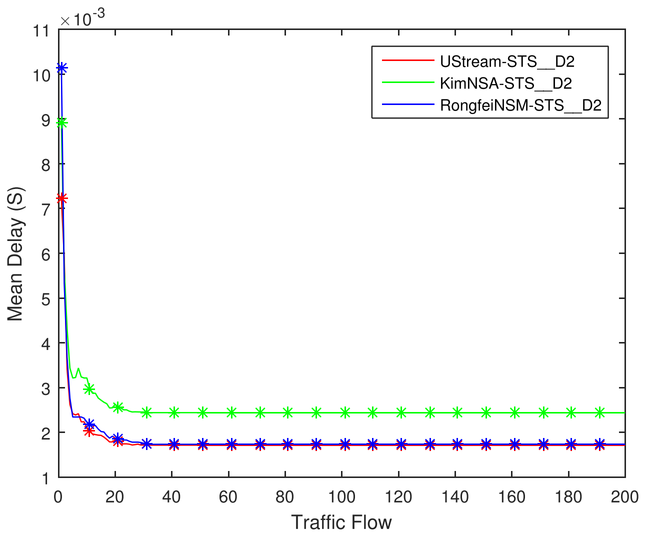


Fig. 12 Slow speed-mean delay(with STS)

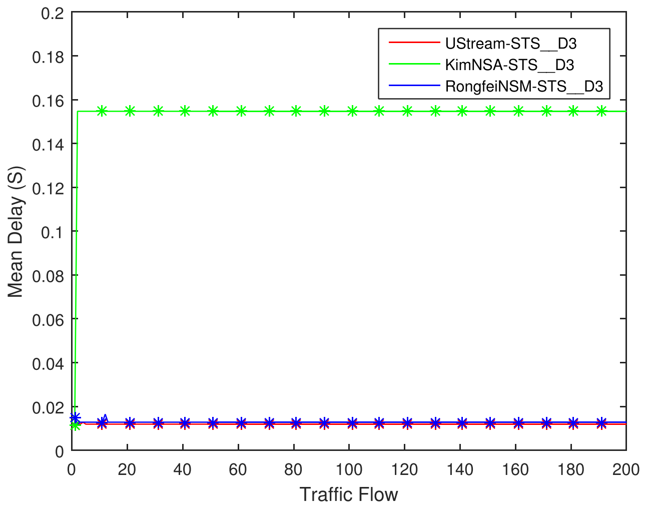


Fig. 14 Extreme slow speed - mean delay(with STS)

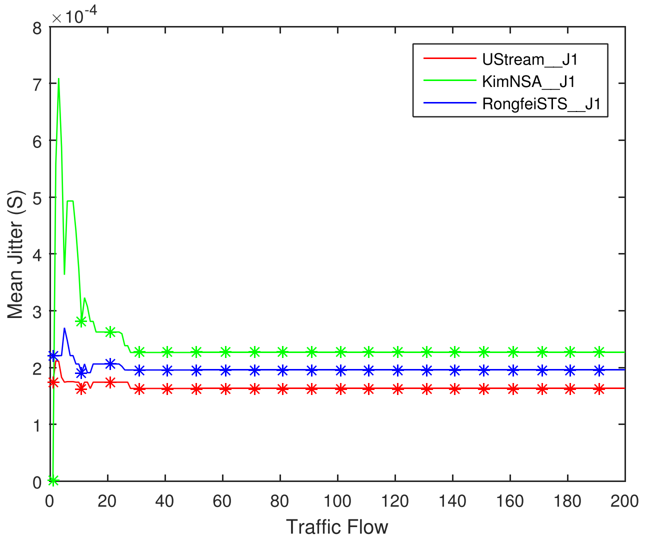


Fig. 15 Average speed - mean jitter

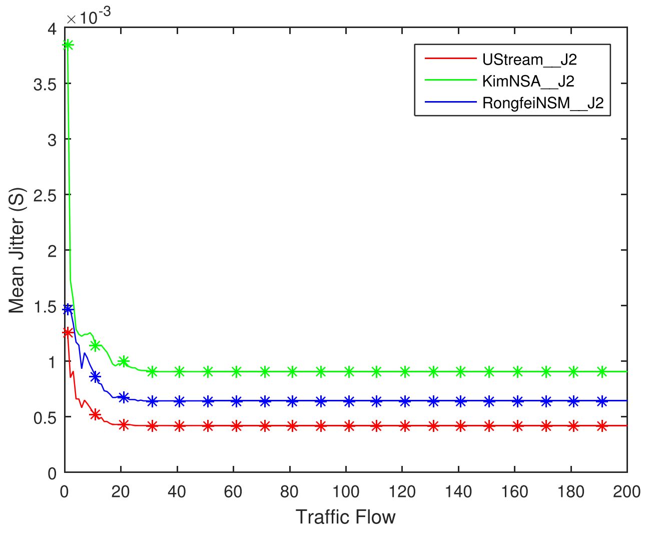


Fig. 17 Slow speed - mean jitter

protocols with STS inclusive achieved lower jitter as shown in Fig. 16. UStream produced significantly low mean jitter of ≈ 0.000185 second while Rongfe-NSM and Kim-NSA achieved a mean jitter of ≈ 0.00021 second and ≈ 0.00061 seconds respectively. However, we observed high level of instability at the beginning of the simulation for the schemes as displaced in Fig. 16.

For the slow speed case study, the result of mean jitter for the three streaming protocols were compared as shown in Fig. 17. The results showed that the UStream accomplished a minimal jitter bounded at ≈ 0.0004 second as against ≈ 0.0007 and ≈ 0.0009 for Rongfe-NSM and Kim-NSA respectively. Figure 18 showed the mean jitter results for the three streaming models with STS inclusive. The graph shows that the UStream outperforms with a steady mean jitter of ≈ 0.00035 seconds as against ≈ 0.00075 seconds and ≈ 0.00085 seconds for Rongfe-NSM and Kim-NSA respectively.

The results presented in Fig. 19 shows an increased mean jitter of ≈ 0.005 second as against ≈ 0.054 second for Rongfe-NSM and ≈ 0.056 for Kim-NSA. This indicates that UStream achieved maximum reduction of 91.1% for extreme slow speed. The combination of STS algorithm with the three schemes reduced the mean jitter for the extreme slow speed with maximum performance of 87.5% when compared with existing scheme(Kim-NSA) as depicted in Fig. 20.

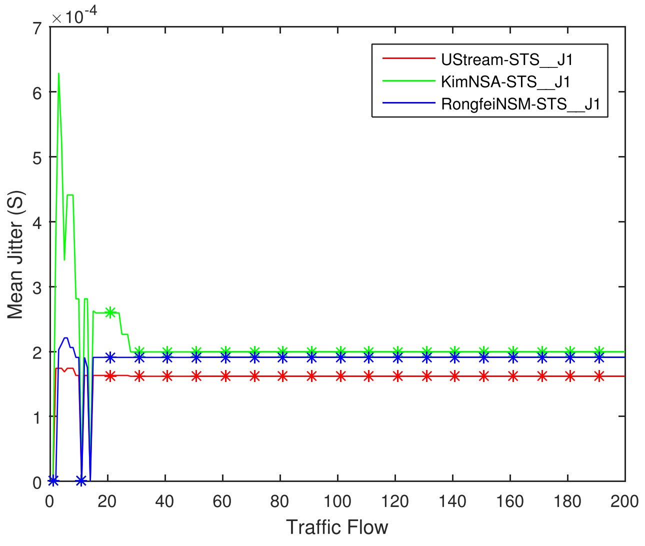


Fig. 16 Average speed - mean jitter(with STS)

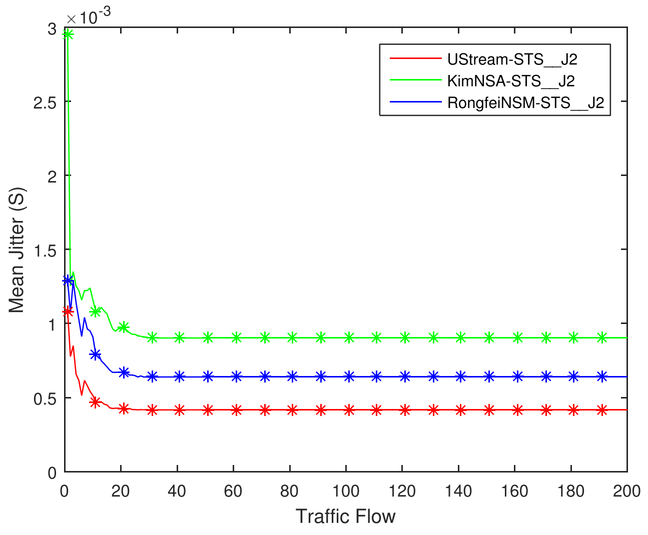


Fig. 18 Slow speed - mean jitter(with STS)

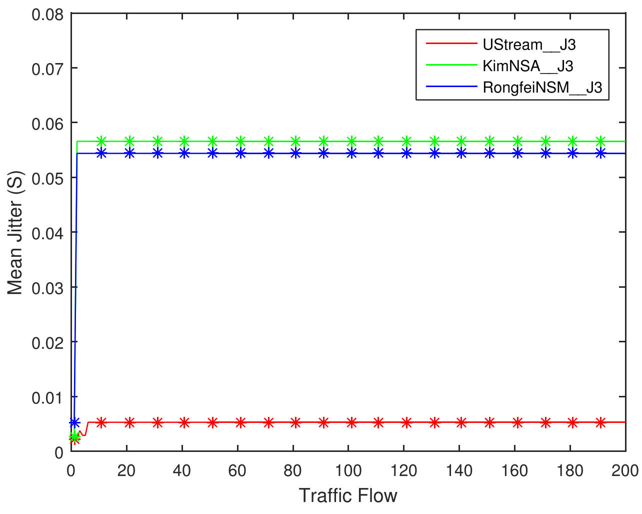


Fig. 19 Extreme slow speed - mean jitter

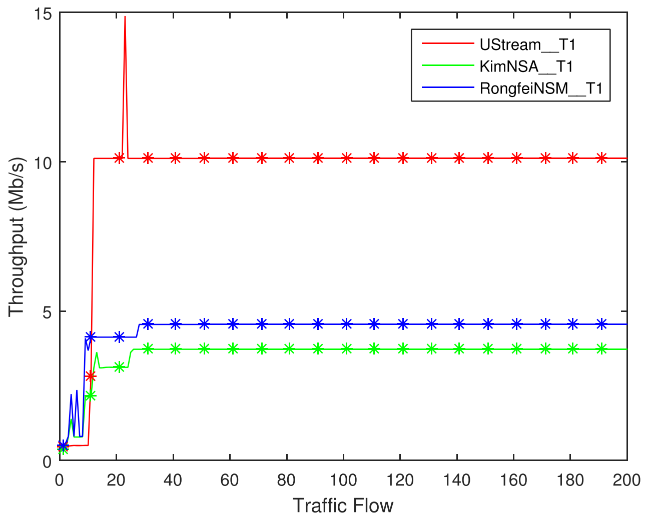


Fig. 21 Average speed - throughput

6.4 Simulation results - throughput

The results of throughput for all scenarios are presented in this section. Figure 21 shows the throughput results in Mbps for average speed category for all the streaming protocol. Kim-NSA achieved an average throughput of 3.5 Mbps and Rongfe-NSM produced an average throughput of 4.5 Mbps. The proposed UStream protocol performs better than the existing protocols with average throughput of 10.1 Mbps, a sudden increase to maximum throughput of 14.8 Mbps for UStream only occurred at a point during the simulation. This simulation result also revealed that the throughput at the beginning of the streaming session is zero which confirms the instability behaviour observed in the mean delay and jitter results. Although, when the proposed scheme and the existing schemes were combined with STS as shown in Fig. 22, higher average throughput is observed. Kim-NSA increased to 4 Mbps and Rongfe-NSM increased to 4.8 Mbps. The UStream achieved average throughput of 15 Mbps and a sudden increase to maximum throughput of 27 Mbps at similar point is as well observed.

In the slow speed category, the throughput of all the streaming protocols with STS and without STS produced relatively similar results. The graph as shown in Fig. 23 shows that Kim-NSA and Rongfe-NSM achieved closely related average throughput of $\approx$ 0.2 Mbps while UStream produced a steady average throughput of $\approx$ 0.3 Mbps. The results revealed that UStream outperforms the existing schemes with increment by 50% ; this proves that the proposed streaming protocol is suitable for transmission of live video in low capacity networks. The throughput results for extreme slow speed also produced similar values as shown in Fig. 24. Furthermore, all the three protocols

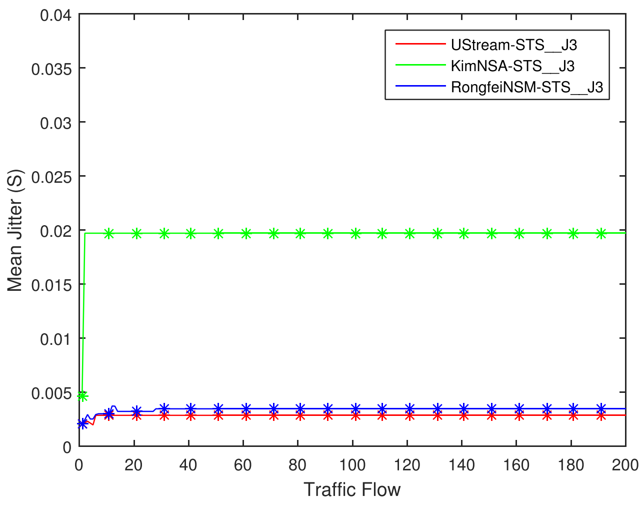


Fig. 20 Extreme slow speed - mean jitter(with STS)

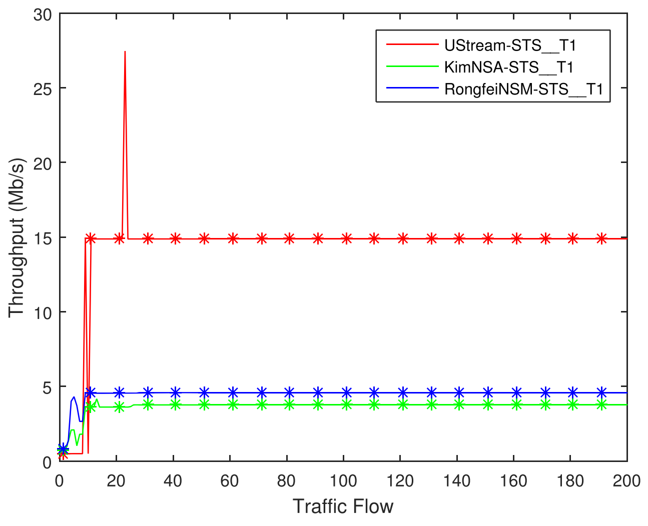


Fig. 22 Average speed - throughput(with STS)

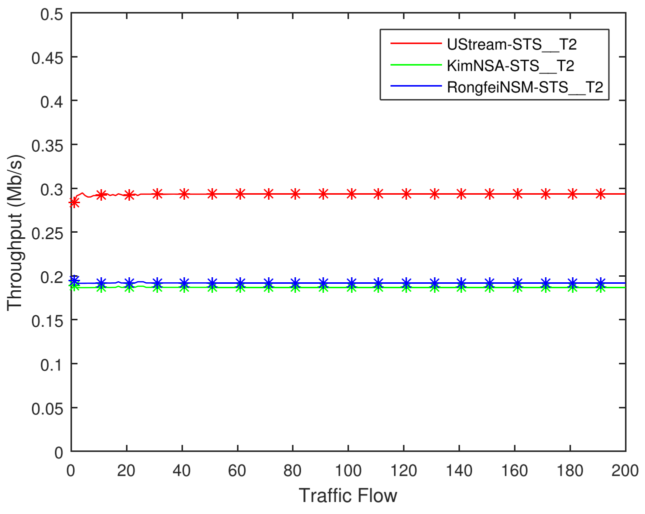


Fig. 23 Slow speed - throughput

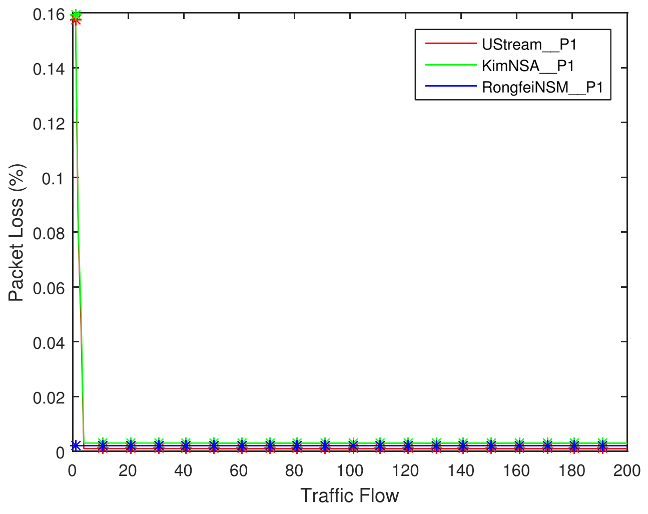


Fig. 25 Average speed - packet loss ratio

achieved closely related values for average throughput of ≈ 0.12 Mbps for Rongfe-NSM and UStream as against ≈ 0.11 Mbps for Kim-NSA. It was also observed that Rongfe-NSM achieved initial throughput of ≈ 0.18 Mbps, this indicates that Rongfe-NSM performs better than UStream in this scenario.

6.5 Simulation results - packet loss ratio

The packet loss ratio(PLR) generates similar results for all the three categories when tested with STS and without STS. The packet loss results for the average speed as shown in Fig. 25; it was observed from the results that all the protocols achieved PLR of ≈ 0.003%. However, initial packet loss of 0.16% is observed for Kim-NSA and UStream before it gradually drops to 0.003% and ≈ 0.0025

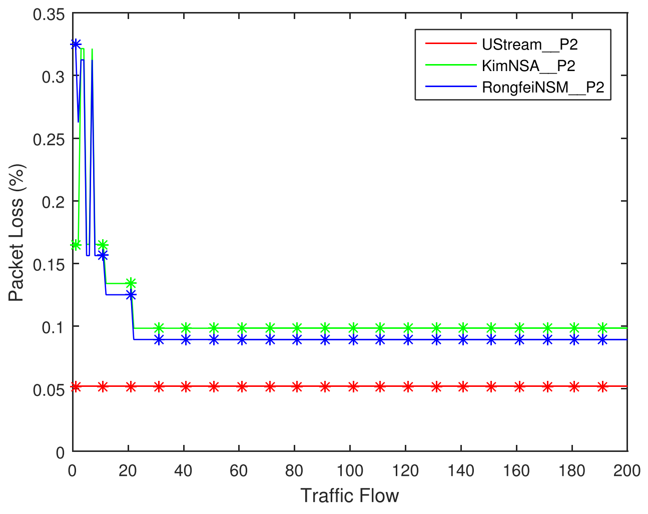


Fig. 26 Slow speed - packet loss ratio

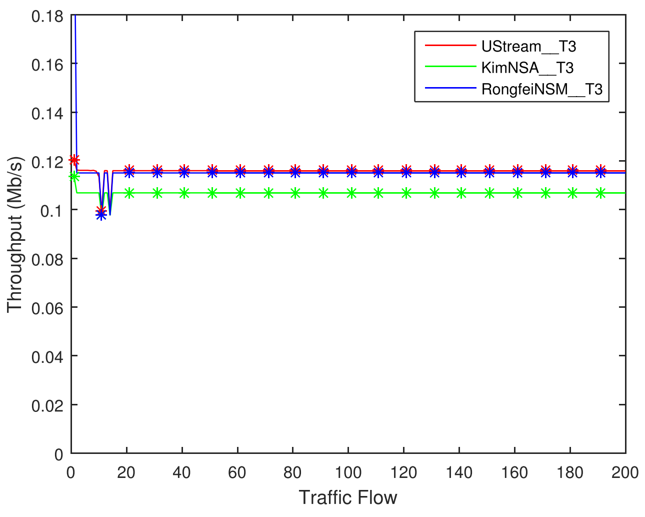


Fig. 24 Extreme slow speed - throughput

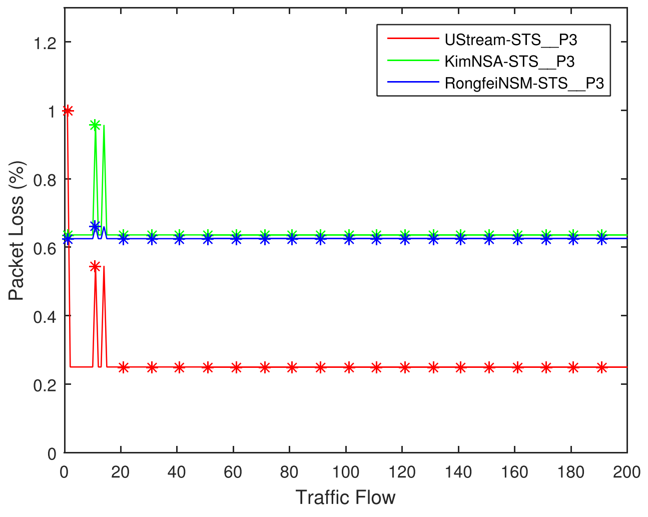


Fig. 27 Extreme slow speed - packet loss ratio

respectively. The slow speed PLR results as given in Fig. 26 showed that UStream maintained steady PLR of 0.05% as against unsteady PLR of (0.09- 0.35%)and (0.1-0.35%) for Rongfe-NSM and Kim-NSA respectively. The extreme slow speed PLR results as shown in Fig. 27 produced lower limit of 0.25% and upper limit of 1% for UStream protocol. The existing protocols achieved bounded PLR of $\approx$ 0.6%, however, Kim-NSA experienced sudden increase at some points with maximum at $\approx$ 0.9%.

7 Conclusion

This work presents an adaptive P2P streaming protocol for effective live video transmission deployable in fast and slow speed networks. The protocol is modeled using scheduling mechanism to manage effective flow of video frames in the network. Additionally, the concept of ultra-metric and spanning tree algorithms were adopted in the formulation of a hybrid overlay topology for fair distribution of resources among peers in P2P streaming systems while reducing waiting time of incoming peers and ensuring provision of alternate routes for peers. The proposed streaming protocol was compared with existing protocols (Rongfe-NSM and Kim-NSA) through extensive simulation within TCP/IP suite in NS3 taking into consideration the existence of network access bottleneck.

The simulation results revealed that our scheme is capable of producing a reliable and better result in comparison with the existing protocols. It is obvious that the proposed protocol has potentials to outperform the existing protocols considering the following indices - minimal start-up delay, lower jitter, reduced end-to-end delay and increased throughput. However, the proposed scheme faired equally in packet loss ratio results with existing protocols. The experimental results demonstrates that the new streaming protocol is suitable for peer-to-peer live streaming system under flash crowd and peer churn situations. It could also be stated that streaming protocol performed efficiently when deployed in low-capacity network.

However, there are still further works to be done to extend the frontier of this research. The proposed scheme can further be refined to further maximise throughput and minimise start-up delay as well as designing a data compression technique suitable for UStream. The new streaming protocol can also be extended for cloud computing, virtual and augmented realities and the 5G network.

Acknowledgements This research is supported by 2014/2015 Tertiary Education Trust Fund (TeTFund) Academic Staff Training and Development (AST&D) Grant of Federal University of Agriculture, Abeokuta, Nigeria.

References

1. van der Schaar M, Chou PA (2007) Multimedia networking and communication: principles and challenges. In: Multimedia over IP and wireless networks. Elsevier, pp 3–10
2. Petrocco R, Pouwelse J, Epema DHJ (2012) Performance analysis of the libswift p2p streaming protocol. In: IEEE 12th International conference on peer-to-peer computing. Tarragona, pp 103–114
3. (2011). Cisco: Cisco visual networking index: Forecast and methodology, 2010-2015. white paper, http://www.cisco.com. Date accessed : 04/04/2014
4. Anjum N, Karamshuk D, Shikh-Bahaei M, Sastry N (2017) Survey on peer-assisted content delivery networks. Comput Netw 116:79–95
5. (2017). Cisco: Cisco visual networking index: Global mobile data traffic forecast 2015-2020. white paper, http://www.cisco.com/c/en/us/solutions/collateral/serviceprovider/visual-networking-index-vni/mobile-whitepaper-c11-520862.html. Date accessed : 02/07/2017
6. Meng X, Tsang PS, Lui KS (2013) Analysis of distribution time of multiple files in a p2p network. Comput Netw 57(15):2900–2915
7. Li B, Wang Z, Liu J, Zhu W (2013) Two decades of internet video streaming: A retrospective view. ACM Trans Multimed Comput Commun Appl 9(1s):33:1–33:20
8. Marza V, Dehghan M, Akbari B (2015) A new peer-to-peer topology for video streaming based on complex network theory. J Syst Sci Complex 28(1):16–29
9. Huang S, Izquierdo E, Hao P (2017) Adaptive packet scheduling for scalable video streaming with network coding. J Vis Commun Image Represent 43:10–20
10. Jinbo S, Alghazawy BA, Fujita S (2017) Batch-based flash crowd relaxation in cloud-assisted p2p live streaming. In: Proceedings of the 18th IEEE/ACIS International conference on software engineering, artificial intelligence, networking and parallel/distributed computing. IEEE, pp 95–100
11. Safara F, Souri A, Baker T, Al Ridhawi I, Aloqaily M (2020) Prinergy: a priority-based energy-efficient routing method for iot systems. J Supercomput, pp 1–18
12. Ciubotaru B, Muntean GM (2013) Advanced network programming - principles and techniques, chap. 2. Springer, pp 3–9. https://doi.org/10.1007/978-1-4471-5292-7
13. Ali QI, Abdul-Jabbar JM et al (2018) Design and implementation of real-time voice streaming evaluation platform over wireless sensor network (vowsn). In: 2018 International Conference on Advanced Science and Engineering (ICOASE). IEEE, pp 233–238
14. Azadmanesh M, Mahdavi M, Ghahfarokhi BS (2019) A reliable and efficient micro-protocol for data transmission over an rtp-based covert channel. Multimedia Systems, pp 1–18
15. Forouzan B, Fegan S (2006) Data Communication and Networking, Special Indian Edition 2006, vol 15, 4th edn., chap. 24, pp. 761–793. Tata McGraw Hill Education Private Limited, 7, West Patel Nagar, New Delhi 110008
16. Ramzan N, Park H, Izquierdo E (2012) Video streaming over p2p networks: Challenges and opportunities. Elsevier Signal Processing: Image Communication 27(2012):401–411
17. Shen Z, Luo J, Zimmermann R, Vasilakos AV (2011) Peer-to-peer media streaming: Insights and new developments. IEEE Journal on Selected Areas in Communications 99(12):2089–2109
18. Hareesh K, Manjaiah D (2011) Peer-to-peer live streaming and video on demand design issues and its challenges. International Journal of Peer to Peer Networks 2(4)

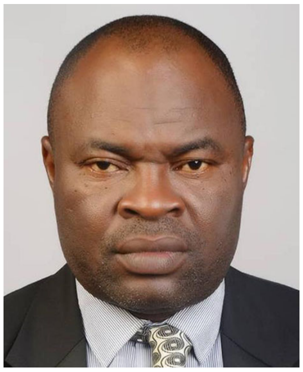
19. Medjiah S, Ahmed T, Boutaba R (2014) Avoiding quality bottlenecks in p2p adaptive streaming. IEEE Journal on Selected Areas in Communications 32(2):734–745
20. Hoßfeld T, Schatz R, Biersack E, Plissonneau L (2013) Internet video delivery in youtube: from traffic measurements to quality of experience. In: Data traffic monitoring and analysis. Springer, Berlin, pp 264–301
21. Adhikari VK, Guo Y, Hao F, Varvello M, Hilt V, Steiner M, Zhang Z (2012) Unreeling netflix: Understanding and improving multi-cdn movie delivery. In: IEEE International conference on computer comminication, pp 1620–1628
22. Hei X, Liang C, Liang J, Liu Y, Ross KW (2007) A measurement study of a large-scale p2p iptv system. IEEE Trans Multimed 9(8):1672–1687
23. Cheng B, Liu X, Zhang Z, Jin H, Stein L, Liao X (2008) Evaluation and optimization of a peer-to-peer video-on-demand system. J Syst Archit 54(7):651–663
24. Dunaytsev R, Moltchanov D, Koucheryavy Y, Strandberg O, Flinck H (2012) A survey of p2p traffic management approaches: best practices and future directions. J Internet Eng 5(1):318–330
25. Zhang X, Hassanein H (2012) Distributed optimization of p2p live streaming overlays. Elsevier Comput Commun 35(2012):1893–1901
26. Kreitz G, Niemela F (2010) Spotify–large scale, low latency, p2p music-on-demand streaming. In: 2010 IEEE Tenth International Conference on Peer-to-Peer Computing (P2P). IEEE, pp 1–10
27. Jiang JR, Hung CW, Wu JW (2010) Bandwidth-and latency-aware peer-to-peer instant friendcast for online social networks. In: 2010 IEEE 16th International conference on Parallel and Distributed Systems (ICPADS). IEEE, pp 829–834
28. e Oliveira JF, Cunha Í., Miguel EC, Rocha MV, Vieira AB, Campos SV (2013) Can peer-to-peer live streaming systems coexist with free riders? In: 2013 IEEE Thirteenth International Conference on Peer-to-Peer Computing, pp. 1–5. IEEE
29. Donnet B, Friedman T (2007) Internet topology discovery: a survey. Communications Surveys and Tutorials. IEEE 9(4):56–69
30. Duraisamy A, Sathiyamoorthy M (2013) Mesh based peer to peer live video streaming using ant algorithm. Int J Eng Adv Technol 2(3):375–380
31. Olayiwola O, Oluwatope A (2014) Packet-level simulation of real-time video in slow-speed environment using ns-3. In: Proceeding of the African Conference, Bostwana, pp 148–153. International Association of Science and Technology for Development
32. 2015 broadband progress report and notice of inquiry on immediate action to accelerate deployment. https://www.theverge.com/2015/1/29/7932653/fcc-changed-definition-broadband-25mbps. Date accessed : 11/08/2020
33. 2020 broadband deployment report. https://docs.fcc.gov/public/attachments/FCC-20-50A1.pdf. Date accessed : 11/08/2020
34. Nigeria has the highest number of internet users in africa. http://www.konbini.com/ng/lifestyle/nigeria-highest-number-internet-users-africa. Date accessed: 2017-10-04
35. Report E (2015) More Nigerians watch videos on smartphone, tablet, laptop. https://www.vanguardngr.com/2015/09/more-nigerians-watch-videos-on-smartphone-tablet-laptop-ericsson-report. Date accessed : 2016-10-10
36. Ojo O, Oluwatope A, Ogunsola O (2015) Ustream: Ultra-metric spanning overlay topology for peer-to-peer streaming systems. In: 2015 IEEE International Symposium on Multimedia (IEEEISM-2015), Miami, Florida, USA. IEEE, pp 601–604
37. Ojo O, Oluwatope A, Ajadi S (2017) Dynamical analysis of an internet-based video system. IFAC-PapersOnLine 50(2):221–226
38. Ojo OE, Oluwatope A, Ajadi SO (2018) Formal verification of a peer-to-peer streaming protocol
39. Ren D, Li YT, Chan SH (2008) On reducing mesh delay for peer-to-peer live streaming. In: The 27th Conference on computer communications. IEEE, pp 1058–1066
40. Small T, Liang B, Li B (2006) Scaling laws and tradeoffs in peer-to-peer live multimedia streaming. In: Proceedings of the 14th annual ACM international conference on Multimedia. ACM, pp 539–548
41. Seibert J, Zage D, Fahmy S, Nita-Rotaru C (2008) Experimental comparison of peer-to-peer streaming overlays: An application perspective. In: 2008 33rd IEEE Conference on Local Computer Networks (LCN). IEEE, pp 20–27
42. Liu S, Zhang-Shen R, Jiang W, Rexford J, Chiang M (2008) Performance bounds for peer-assisted live streaming. In: ACM SIGMETRICS Performance Evaluation Review, vol 36. ACM, pp 313–324
43. Biskupski B, Schiely M, Felber P, Meier R (2008) Tree-based analysis of mesh overlays for peer-to-peer streaming. In: Distributed Applications and Interoperable Systems. Springer, pp 126–139
44. Liu J, Rao SG, Li B, Zhang H (2008) Opportunities and challenges of peer-to-peer internet video broadcast. Proc IEEE 96(1):11–24
45. Goh CY, Yeo HS, Lim H, Hoong PK, Lim J, Tan K (2015) A comparative study of tree-based and mesh-based overlay p2p media streaming. International Journal of Multimedia and Ubiquitous Engineering 8(4):97–106
46. Magharei N, Rejaie R, Guo Y (2007) Mesh or multiple-tree: A comparative study of live p2p streaming approaches. In: Proceedings of the 26th IEEE International Conference on Computer Communications. IEEE, pp 1424–1432
47. AlTuhafi AW, Ramadass S, Chong YW (2013) Concepts and types of peer-to-peer network topology for live video streaming. In: 2013 IEEE International Conference on RFID-Technologies and Applications (RFID-TA). IEEE, pp 1–4
48. Liu Y, Guo Y, Liang C (2008) A survey on peer-to-peer video streaming systems. Peer-to-peer Networking and Applications 1(1):18–28
49. Magharei N, Rejaie R, Guo Y (2011) Incorporating contribution-awareness into mesh-based peer-to-peer streaming systems. Peer-to-Peer Networking and Applications 4(3):231–250
50. Abboud O, Pussep K, Kovacevic A, Mohr K, Kaune S, Steinmetz R (2011) Enabling resilient p2p video streaming: survey and analysis. Multimedia Systems 17(3):177–197
51. Ghoshal J, Xu L, Ramamurthy B, Wang M (2007) Network architectures for live peer-to-peer media streaming. CSE Technical reports, p 78
52. Hei X, Liu Y, Ross KW (2008) Iptv over p2p streaming networks: the mesh-pull approach. IEEE Communications Magazine 46(2)
53. Chen Y, Farley T, Ye N (2004) Qos requirements of network applications on the internet. Information Knowledge Systems Management 4(1):55–76
54. Hareesh K, Manjaiah D (2013) Quality of service in peer to peer video on demand system using v chaining mechanism. Int J Comput Info Technol 2(1):109–117
55. Feng C, Li B (2008) On large-scale peer-to-peer streaming systems with network coding. In: Proceedings of the 16th ACM international conference on Multimedia. ACM, pp 269–278
56. Krishnan SS, Sitaraman RK (2013) Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. IEEE/ACM Transactions on Networking (TON) 21(6):2001–2014

57. Wu PJ, Hwang JN, Lee C, Gau CC, Kao HH (2009) Eliminating packet loss accumulation in peer-to-peer streaming systems. IEEE Trans Circuits Sys Vid Technol 19(12):1766–1780
58. Lo CW, Lin C, Chen YC, Yu JY (2011) Contribution-based peer selection for packet protection for p2p video streaming over mesh-based networks. In: 18th IEEE International Conference on Image Processing (ICIP). IEEE, pp 2233–2236
59. Lo CW, Lin C, Chen YC, Yu JY (2011) A packet loss estimation model and its application to reliable mesh-based p2p video streaming. In: IEEE International Conference on Multimedia and Expo (ICME). IEEE, pp 1–6
60. Akbari B, Rabiee HR, Ghanbari M (2008) Packet loss in peer-to-peer video streaming over the internet. Multimedia Systems 13(5-6):345–361
61. Goyal P, Vin HM, Shen C, Shenoy PJ (1996) A reliable, adaptive network protocol for video transport. In: Proceedings of the Fifteenth Annual Joint Conference of the IEEE Computer Societies, vol 3. IEEE, pp 1080–1090
62. Noh J, Baccichet P, Mavlankar A, Girod B (2008) Un-leeching p2p streaming by active overlay management. In: Global telecommunications conference. IEEE, pp 1–5
63. Zhang M, Xiong Y, Zhang Q, Sun L, Yang S (2009) Optimizing the throughput of data-driven peer-to-peer streaming. IEEE Transactions on Parallel and Distributed systems 20(1):97–110
64. Farid F, Shahrestani S, Ruan C (2013) Qos analysis and evaluations: Improving cellular-based distance education. In: Proceedings of the IEEE 38th Conference on local computer networks. IEEE, pp 17–23
65. (2008). ITU-T: Quality of experience requirements for iptv services. Recommendation ITU-T G.1080
66. Locher T, Meier R, Schmid S, Wattenhofer R (2007) Push-to-pull peer-to-peer live streaming. In: Distributed Computing. Springer, pp 388–402
67. Fallica B, Lu Y, Kuipers F, Kooij R, Van Mieghem P (2008) On the quality of experience of sopcast. In: The Second International Conference on Next Generation Mobile Applications, Services and Technologies, pp. 501–506. IEEE
68. Fallica B, Lu Y, Kuipers F, Kooij R, Mieghem PV (2008) On the quality of experience of sopcast. In: The Second International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST). IEEE, pp 501–506
69. (2007). SopCast: Sopcast - free p2p internet tv. http://www.sopcast.org. Date accessed : 15/06/2018
70. Kao HH, Lee C, Kao YC, Wu PJ (2016) I2cc: Interleaving two-level cache with network coding in peer-to-peer vod system. J Netw Comput Appl 60:180–191
71. Hammami C, Jemili I, Gazdar A, Belghith A, Mosbah M (2014) Hybrid live p2p streaming protocol. Procedia Computer Science 32:158–165
72. Ono K, Zhygmanovsky A, Matsumoto N, Yoshida N (2014) Resilient live-streaming with dynamic reconfiguration of p2p networks. In: Proceedings of the Sixth International Conference on Emerging Network Intelligence, pp 1–6. Rome, Italy
73. Ono K, Zhygmanovsky A, Matsumoto N, Yoshida N (2016) Motif-based qos-aware dynamic optimization of p2p streaming networks. British Journal of Mathematics and Computer Science 17(2):1–20
74. Chen X, Ren S, Wang H, Zhang X (2005) Scope: Scalable consistency maintenance in structured p2p systems. In: 24th Annual Joint Conference of the IEEE Computer and Communications Societies, vol 3. IEEE, pp 1502–1513
75. Jagadish HV, Ooi BC, Vu QH (2005) Baton: A balanced tree structure for peer-to-peer networks. In: Proceedings of the 31st international conference on very large data bases. VLDB Endowment, pp 661–672
76. Jin X, Yiu W, Chan GSH, Wang Y (2007) On maximizing tree bandwidth for topology-aware peer-to-peer streaming. IEEE Transactions on Multimedia 9(8):1580–1592
77. Zulhasnine M, Huang C, Srinivasan A (2012) Towards an effective integration of cellular users to the structured peer-to-peer network. Peer-to-Peer Networking and Applications 5(2):178–192
78. Noh J, Girod B (2012) Robust mobile video streaming in a peer-to-peer system. Signal Process Image Commun 27(5):532–544
79. Maani E, Chen Z, Katsaggelos AK (2012) A game theoretic approach to video streaming over peer-to-peer networks. Signal Process Image Commun 27(5):545–554
80. Zhang X, Liu J, Li B, Yum T (2005) Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming. In: 24th Annual joint conference of the IEEE computer and communications societies, vol 3. IEEE, pp 2102–2111
81. Liao X, Jin H, Liu Y, Ni LM, Deng D (2006) Anysee: Peer-to-peer live streaming. In: 25th IEEE International conference on computer communications, pp 1–10
82. Liao X, Jin H, Liu Y, Ni LM (2007) Scalable live streaming service based on interoverlay optimization. IEEE Transactions on Parallel and Distributed Systems 18(12):1663–1674
83. Pianese F, Perino D, Keller J, Biersack EW (2007) Pulse: an adaptive, incentive-based, unstructured p2p live streaming system. IEEE Trans Multimed 9(8):1645–1660
84. Carra D, Lo Cigno R, Biersack EW (2007) Graph based analysis of mesh overlay streaming systems. IEEE Journal on Selected Areas in Communications 25(9):1667–1677
85. Hossain T, Cui Y, Xue Y (2009) Rate distortion optimization for mesh-based p2p video streaming. In: IEEE International Conference on Communications. IEEE, pp 1–6
86. Li R, Wu Q, Lin Y, Lu X, Wang Z (2010) On topology construction in layered p2p live streaming networks. In: Network Operations and Management Symposium. IEEE, pp 599–606
87. Zhao C, Lin X, Wu C (2011) The streaming capacity of sparsely-connected p2p systems with distributed control. In: Proceedings of IEEE International conference on computer communications. IEEE, pp 1449–1457
88. Ren D, Wong WK, Chan SHG (2012) Toward continuous push-based p2p live streaming. In: Global communications conference. IEEE, pp 1969–1974
89. Zhang J, Zhang X, Yang C (2018) Towards the multi-request mechanism in pull-based peer-to-peer live streaming systems. Comput Netw 138:77–89
90. Barekatain B, Khezrimotlagh D, Maarof MA, Quintana AA, Cabrera AT (2017) Gazelle: an enhanced random network coding based framework for efficient p2p live video streaming over hybrid wmns. Wirel Pers Commun 95(3):2485–2505
91. Kim E, Kim J, Lee C (2019) Efficient neighbor selection through connection switching for p2p live streaming. J Ambient Intell Humaniz Comput 10(4):1413–1423
92. Xie S, Li B, Keung GY, Zhang X (2007) Coolstreaming: Design, theory, and practice. IEEE Trans Multimed 9(8):1661–1671
93. Li B, Xie S, Qu Y, Keung GY, Lin C, Liu J, Zhang X (2008) Inside the new coolstreaming: Principles, measurements and performance implications. In: INFOCOM 2008. The 27th Conference on Computer Communications. IEEE, pp 1031–1039
94. Wang F, Xiong Y, Liu J (2010) mtreebone: A collaborative tree-mesh overlay network for multicast video streaming. IEEE Transactions on Parallel and Distributed Systems 21(3):379–392
95. Moshref M, Motamedi R, Rabiee HR, Khansari M (2010) Layeredcast-a hybrid peer-to-peer live layered video streaming protocol. In: 2010 5th International Symposium on Telecommunications (IST). IEEE, pp 663–668

96. Pal K, Govil M, Ahmed M (2015) A new hybrid approach for overlay construction in p2p live streaming. In: 2015 International conference on advances in computing, communications and informatics (ICACCI). IEEE, pp 431–437
97. Rückert J., Richerzhagen B, Lidanski E, Steinmetz R, Hausheer D (2015) Topt: Supporting flash crowd events in hybrid overlay-based live streaming. In: IFIP Networking Conference (IFIP Networking), 2015. IEEE, pp 1–9
98. Maheswari BU, Sudarshan T (2016) Reputation based mesh-tree-mesh cluster hybrid architecture for p2p live streaming. In: 2016 3rd International Conference on Devices, Circuits and Systems (ICDCS). IEEE, pp 240–243
99. Demirci S, Yardimci A, Sayit M, Tunali ET, Bulut H (2017) A hierarchical p2p clustering framework for video streaming systems. Computer Standards & Interfaces 49:44–58
100. Maheswari BU, Ramesh T (2018) An improved delay-resistant and reliable hybrid overlay for peer-to-peer video streaming in wired and wireless networks. IEEE Access 6:56539–56550
101. Rongfei M (2019) Super node selection algorithm combining reputation and capability model in p2p streaming media network. Pers Ubiquit Comput 23(3-4):435–442
102. Liu Z, Murray N, Lee B, Fallon E, Qiao Y (2018) Mvp2p: Layer-dependency-aware live mvc video streaming over peer-to-peer networks. Signal Process Image Commun 65:173–186
103. Olayiwola O, Oyewo D, Oluwatope A, Aderounmu A, Adagunodo R (2012) Double-buffer traffic shaper modelling for multimedia applications in slow speed network. Issues in Informing Science and Information Technology 9:361–368
104. Seeling P, Reisslein M (2011) Video transport evaluation with h. 264 video traces. IEEE Communications Surveys & Tutorials 14(4):1142–1165. http://trace.eas.asu.edu/videotraces2/h265/. Date accessed : 06/07/2020

Oluwafolake E. Ojo received B.Sc. degree in Computer Science from Olabisi Onabanjo University, Nigeria in 2007. She also obtained M.Sc. degree and Ph.D. degree in Computer Science from Obafemi Awolowo University, Nigeria in 2012 and 2017 respectively. She is a Lecturer in the Department of Computer Science, Federal University of Agriculture, Abeokuta, Nigeria, where she teaches and conducts research. Her research interests include multimedia computing, network management and protocol modeling & simulation. She has published several articles in journals and international conferences. She is a professional member of Association of Computer Machinery (ACM) and Nigeria Computer Society.

Ayodeji O. Oluwatope received the BSc. Degree (Hons) in Computer Engineering in 1995, MSc and PhD. Degrees in computer science in 2003 and 2008 respectively from the Obafemi Awolowo University, Ile-Ife, Nigeria. He is a member of the Nigerian Computer Society, NCS, Nigeria Society of Engineers, (NSE), Association of Computing Machinery and Registered Engineer. He is currently a Professor of Computer Science and Engineering with specialties in data communication and networking with a particular emphasis on modeling, simulation and performance. His recent research activities include embedded system, IoT and 5G.



Suraju O. Ajadi is a professor of Applied Mathematics. He obtained a B.Sc., M.Sc. and Ph.D. degrees in 1994, 1999 and 2005 respectively–all from Obafemi Awolowo University, Ile-Ife, Nigeria. He was employed at the same institution as a Graduate Assistant in 1996. He rose through the ranks and was promoted Professor of Applied Mathematics by the authorities of Obafemi Awolowo University, Ile-Ife, Nigeria in 2012. He is a teacher and researcher in undergraduate and postgraduate activities. He has supervised a few undergraduate and postgraduate students' theses. He has published and also served as reviewers of some reputable journal outlets.