



A fuzzy based hierarchical flash crowd controller for live video streaming in P2P network

Md. Sarfaraj Alam Ansari¹ · Kunwar Pal² · Prajval Govil³ · Mahesh Chandra Govil¹

Received: 30 April 2022 / Accepted: 15 February 2023 / Published online: 1 March 2023
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract

A Peer-to-peer (P2P) network system has proved to be a popular and efficient approach for large-scale live video streaming and file sharing over the Internet. Such a system is always challenged by various issues and requires to be monitored and controlled for significant performance. Among them, one of the major performance issues in P2P networks is the flash crowd. When many users switch to a popular channel in a short span of time, the P2P system suffers from a serious join congestion problem, called the flash crowd problem. In such situations, systems can scale only up to a limit, both by available surplus bandwidth and intense competition among peers for their start-up, and hence, it not only decreases the efficiency but also creates a blockage of services for connected peers. Therefore, management of the flash crowd has been important and becoming paramount with the increased use of online services and the proliferation of various technologies like IoT, Edge & Mobile Computing, Crypto Currencies, and Torrenz-like services. A hierarchy-based admission control model with multiple threshold layers is proposed to mitigate the flash crowd in the P2P network paradigm. A fuzzy logic-based prediction mechanism is proposed to predict the control scheme applicable to the requested peers based on the parameters such as buffer size, bandwidth, etc. In case of the occurrence of the flash crowd, the model applies different control schemes depending upon which threshold layer it is predicted. These include saving the system resources, granting preferences to best-performing peers, dropping requests of worst-performing peers, a slight decrease in service quality, etc. We also characterize the credit score of peers, which denotes their performances. The model is simulated in OMNET++ along with the OverSim framework. Results from the simulation reveal a reduction in request drops, processing delays, and startup delays compared to other management schemes.

Keywords Peer-to-Peer networks · Video streaming · Flash crowd · Fuzzy logic · Selfish peer

Kunwar Pal, Prajval Govil and Mahesh Chandra Govil are authors contributed equally to this work.

✉ Md. Sarfaraj Alam Ansari
sarfaraj@nitsikkim.ac.in
Kunwar Pal
kunwarp@nitj.ac.in
Prajval Govil
prajvalgovil@gmail.com
Mahesh Chandra Govil
govilmc@gmail.com

¹ Department of Computer Science & Engineering,
National Institute of Technology Sikkim, Ravangla,
South Sikkim 737139, Sikkim, India

² Department of Computer Science & Engineering, Dr B
R Ambedkar National Institute of Technology, Street,
Jalandhar 144027, Punjab, India

³ Department of Computer Science & Engineering, JK
Lakshmipat University, Jaipur 302026, Rajasthan, India

1 Introduction

In today's world, the Internet has become the way of life and has penetrated in its every domain. It is nearly impossible to list the applications which make use of the Internet in this era. However, all these applications are data-intensive, and data may be textual, audio or visual, requiring improved techniques to deal with these. Multimedia applications are one of them and have witnessed unprecedented growth in the last few years. A predominance of that is by virtue of different video streaming applications in daily life like games, education, entertainment, security etc. [1–3].

The Cisco study reveals that 80-90% of traffic over the Internet up to 2021 is due to video transmission [4]. This sudden increase in the demand for video has contributed to an exponential increase in the popularity of different media platforms over the Internet. Some of these popular media transmission platforms are YouTube [5], IPTV [6], NetTV

[7], etc. The popularity of these platforms was evaluated by the Alexa ranking system, and YouTube is listed next to Google [8]. All forms of IP video, which are considered video file exchange, video conferencing, Video-on-Demand (VoD), Live video games, and Internet video, are 80-90% of total IP traffic. A significant portion of the Internet traffic is due to multimedia transmission. Demand for multimedia streaming in human life is increasing rapidly. Its not possible for everyone to attend the event live due to different geographical locations, monitoring costs, and feasibility issues. So, it is better to watch the events on the client site, either by using live streaming or video on demand. Justifying the demand of such huge crowd for live video streaming in CDN is facing a lack of resources issues. So, using clients' resources (Peer-to-Peer Network) can resolve this issue. In the present days, peer-to-peer is getting popularity due to its huge application domains like bit-torrent. The same has also opened the path for various vulnerabilities.

The flash crowd [9, 10] may occur when huge number of users arrive and connect to a particular server for services in a short period. It is typically triggered by popular programs that cause a surge of users to join the system at a scheduled time. In P2P systems, users usually need to contact a local tracker server to retrieve a peer list when they switch channels [11]. If a large number of users switch to the same channel in a short time, the server will be overloaded with a large number of requests resulting in requests overflow. Flash Crowd severely affects the network and communication among stable peers. In a P2P network, it highly decreases the performance of the system. Flash crowd causes a huge network traffic which reduces system bandwidth drastically, and as a result system gets out of service. The Fig. 1 shows a pictorial view of flash crowd. The dynamic behaviour of the network during flash crowd is observed from the figure. A peak is formed when the peer arrival rate is very high. There is a considerable time difference between flash crowd and steady state [12].

A number of studies exist to resolve the issue of 'flash crowd' in P2P networks [13–22]. Flash crowd detection

is one of the crucial components of flash crowd handling. Generally, admission control strategies are used to control the flash crowd, and the tracker is responsible for allowing the incoming join requests. The number of join requests is adaptively adjusted according to the network capacity. However, this approach has a few drawbacks i.e. the waiting time is high for some peers to get their turn. In a distributed environment, it is difficult to calculate the network capacity. Some of the popular flash crowd detection approaches uses arrival rate as one of the prime identification methods [23]. For admission control approach uses the bandwidth-aware mechanism. Other approaches like [24] use cache-based system for the admission control system in P2P network while [13–15] uses slot-based and batch-based admission control [19].

Without admission control, it is nearly impossible to handle a flash crowd. Earlier flash crowd detection approaches primarily focus on arrival rate only, while admission control largely focuses on either bandwidth-aware or anomaly-based admission control in P2P live video streaming and blocks the unwanted peers. There needs to be more flash crowd detection using other crucial factors like event criticality, buffer size, available bandwidth and arrival rate of peers.

For admission control on the basis of severity of the flash crowd, a hierarchical admission controller is required. Motivation of peers can also help in increasing the required resources specially in P2P Network instead of just blocking the peers. For considering the above issues, the proposed solution developed a Fuzzy-based Flash Crowd Predictor (FFCP), which considers the various crucial factors, e.g. arrival rate, event-based flash crowd, and remaining bandwidth. Then developed, the hierarchical-based admission controller for handling the population of the flash crowd. The results of the proposed solution are quite motivating, and efficiently handle the flash crowd and improve the overall performance of the network. The admission controller not only blocks the peer but also encourages them to active participation in the network.

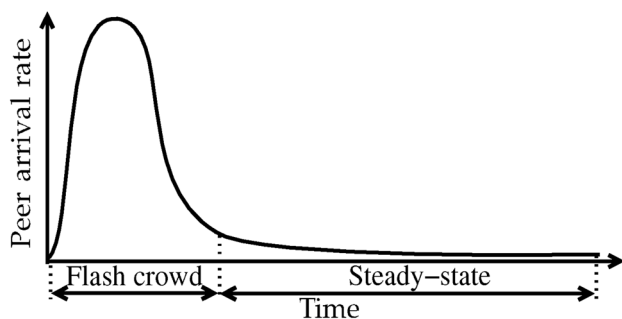


Fig. 1 Graphical view of Flash Crowd (Peer arrival Vs time) [12]

1.1 Research contributions

The contributions made in this paper are summarised as follows:

1. A fuzzy-based Flash Crowd Predictor (FCP) is designed and developed to predict the severity of flash crowd in a P2P network.
2. Fuzzy rules are developed considering the factor - remain bandwidth, buffer size and events-time.
3. A novel algorithm, 'Fuzzy-based Hierarchical Flash Crowd Controller', is proposed to control the flash crowd.

4. Congestion Control Schemes are designed and developed for different modes (Like Underlay, Overlay, Scheduler and Application Mode).

The remaining part of the paper is arranged as follows: Literature survey on flash crowd is discussed in Section 2. Section 3 describes the Proposed Fuzzy-based Flash Crowd Predictor (FFCP) for the identification of flash crowd. Section 4 discusses the Fuzzy-based Hierarchical Flash Crowd Controller (*FHFC*²). Section 5 discusses the criticality of the algorithm. Section 5 discusses the results and discussion. Finally, Section 6 draws the conclusion and future direction of work.

2 Literature survey

The limitation of client-server architecture motivates researchers to move toward peer-to-peer network. The peer-to-peer network can play a significant role to fulfil that demand. Primarily peer-to-peer networks are used for file transfer. Bittorrent [25], Gnutella [26], FastTrack [27] and eDonkey [28] are some of the most popular P2P file sharing systems. A huge amount of work has already been done to improve the performance of the P2P file sharing system. A group-based approach for enhancing the file distribution system is provided by Ma Lingjun et al. [29], a better file replication system can also improve the performance of the network [30]. Prime components of the P2P file system which affects the performance of the networks are overlay, scheduling and selfish peer identification [29, 31] and [32].

Due to the scalability and ease of resource utilization, demand for peer-to-peer networks is increasing in resource-hungry applications like multimedia streaming. Multimedia traffic play a significant role in IP traffic. Out of which, a gigantic amount of the traffic is due to adult video content. A study on xHamster, a popular adult streaming platform, is conducted by Farrelly et al. [33, 34]. A comparative analysis of various user-generated content-based porn sites like Pornhub, xHamster, and YouPorn are provided by Raymond Yu et al. [35]. The results are quite interesting and motivates researchers to explore this area of multimedia streaming. The mobile phone also plays a significant role in Internet traffic. Initially, the data traffic generated by the fixed devices are the primary reasons for Internet traffic but in last few years due to the enhancement of technology, mobile phone also played a significant role in the same. For identification of the exact impact of mobile phones over the Internet traffic a study is conducted by Yo-Der Song et al. in an Academic Web Server [36]. Some traffic analysis studies based on HTTP, like cyberlock is also discussed by Aniket Mahanti et al. [37]. Power-laws are used for analysis the

Internet traffic and further the power-laws how can improve the performance of the Internet-based system [38].

The P2P live streaming system can be viewed as a set of peers that collaborate with each other to disseminate live media transmissions [39, 40]. Certain peers work as a server that encodes the video, splits the video into chunks and starts distribution among the peers. Here, chunk refers to one or more frames [41]. There are many factors that arises in overlay maintenance and media redistribution strategies in the context of P2P media distribution during flash crowd. For example, peer churn, caused by peers joining and leaving the overlay, break partnerships and disrupt media distribution [42]; uncooperative peers, sometimes called selfish peer, which do not contribute to media redistribution, increase resource competition in the overlay [39, 43]. Such approaches often involve sophisticated overlay maintenance [44, 45] or request scheduling [46, 47]. A detailed comparative analysis of peer-to-peer and web traffic is shown by the Naimul Basher et al. Comparative analysis of both traffic is shown with respect to various crucial factors like Flow Size, Flow Inter-arrival time, Flow Duration etc. Presented results shows the growth of peer-to-peer network in data transmission [48]. Existing results on commercial P2P live streaming systems such as UUSee [46, 49] and Cool-Streaming [9] indicate that the performance of such systems is in general acceptable under small-sized flash crowd, but degrades severely under large-sized flash crowds. In the latter case, a lot of users get congested and cannot even watch the program.

A number of studies exist to resolve the issue 'flash crowd' in P2P networks [13–22]. Generally, admission control strategies are used to control the flash crowd, and tracker is responsible for allowing the incoming join requests. It is known that the system can accept a limited number of join requests depending on the system capacity. Thus, scrambling of the upload capacity of existing nodes could be relaxed by allowing a limited number of selected nodes to send a join request to the tracker. The number of join requests is adaptively adjusted according to the network capacity, however, the approach has few drawbacks. The waiting time is long for some of the peers to get their turn. In a distributed environment, it is difficult to calculate the network capacity.

This admission control mechanism is very common to prevent the streaming systems from being overloaded. In the admission control approach, it rejects excessive incoming requests or holds the requests temporarily until the resources appear to be vacated [50]. In a slot-based approach, the peers who arrived newly can be admitted on a per-slot basis. The residual upload bandwidth of the peer is divided according to the steaming rate, and equal number of newly arrived peers will be allowed to admit by the tracker [13–15]. Usually, the end-to-end delay among peers is not known to the tracker server. Estimating the optimal slot length is difficult

since it depends upon the end-to-end delay. Therefore, the estimation of slot length will be a critical task for tracker server since the smaller slots may make the system unstable, while longer slots give larger start-up latency.

Another method is to construct an overlay called a *batch* consisting of newly arrived peers and allow some peers in the batch to connect to the existing overlay. Tracker is responsible for the construction and control of any batch [22]. But it is difficult to design a batch to continue to receive chunks. Further, Flash crowds defines in terms of shock level [19]. Let λ_1 be the rate of arrival of new peer after the flash crowd and λ_0 be the peer arriving rate before flash crowd. The shock level of a flash crowd is defined as the ratio of the peer arrival rate during and before the flash crowd. Then the shock level (v) is equal to $\frac{\lambda_1}{\lambda_0}$. When λ_0 is not specified, a large flash crowd means a high shock level; when λ_0 is given, a large flash crowd means not only a high shock level, but λ_1 is also significantly larger than λ_0 . The highest value of shock level defines the capacity of the system. Chen et al. [19] estimated flash crowd strength as shock level. They examine the flash crowd dynamics by using a fluid-based model. Using shock level and the proper population control measures the waiting time of peers can be made to increase logarithmically.

Liu et al. [14, 20] have discussed the flash crowd dynamics in mesh-based live streaming systems in which a network can scale up to a limit. Many peers can be satisfied in their initial attempts with surplus bandwidth. But peers take time to locate resources, therefore, having available surplus bandwidth is not sufficient. In tree based P2P live streaming, it has a pre-constructed topology. The video content can be retrieved from pre-determined parents. So, the time-scale issue is not there at the tree-based system. Wu et al. [21] construct many trees based on the available surplus bandwidth of the system. In this approach, merit has been calculated based on forwarding capability and waiting time of each peers. Further, it finds the positioning of each peer into different layers of the tree(s) based on merit ranks. It connects peers to a peer in a layer above it and joins the root node of the tree(s) to the stable peer(s). A tracker is responsible for performing the above task. It burdens the tracking server and makes it a potential single point of failure.

The extensive study exhibit that admission control is a common way of dealing with flash crowd in which requests are either simply denied or are accepted based upon some rules defined to support the system. There are various admission control mechanisms which enhance the system's performance up to some extent. However, a complete solution for the detection of the criticality of flash crowd in the P2P network and controlling mechanism for flash crowd is in demand. Realizing this need, a fuzzy-based flash crowd predictor (FCP) is designed and developed to predict the

severity degree of flash crowd in the P2P network. This predictor is used to design and develop a hierarchical approach to manage or control the flash crowd in the P2P network. Detailed discussions on the proposed approaches are given subsequently. Table 1 shows the description of different work conducted on peer-to-peer network with respect to its domain.

3 Proposed fuzzy-based flash crowd predictor (FFCP)

The fuzzy system is popular because of its inherent characteristics such as clarity and simplicity. A fuzzy system is also beneficial when a system is not fully understood, and an approximate output is required quickly. It is suitable for use with a P2P application. This is why we have used fuzzy logic for managing the flash crowd. A comparison of the fuzzy system with other approaches is given in Table 2. A novel approach called fuzzy-based hierarchical flash crowd controller (*FHFC²*) is proposed and presented in the next section.

A Fuzzy-based Flash Crowd Predictor (FFCP) is designed for controller selection. It determines the criticality of flash crowd and provide the information to the Flash Crowd Controller. For evaluating the criticality, different parameters like total remain bandwidth, pending request (Buffer status), event and time window are used. To emulate real-time reflection on a flash crowd controller, all of these inputs to the fuzzy model are provided directly from the network.

Fuzzification, inference engine, fuzzy rule base, and defuzzification are the four essential components of the Fuzzy based Flash Crowd Predictor (FFCP). In the fuzzification process, the fuzzifier takes crisp network inputs and changes it to fuzzy values, a process known as fuzzification. The inference engine is the main part of the fuzzy controller which uses fuzzy rules to apply logical reasoning and make decisions. To determine the criticality of network congestion, the defuzzifier uses specified fuzzy sets and membership functions to find the crisp logic output. Figure 2 depicts the proposed fuzzy-based flash crowd controller model.

3.1 Fuzzification

Fuzzification is a procedure that converts various input variables into linguistic variables in order to calculate the membership degree. Data compression can be used to express linguistic variables. A set of numerical values can be used to represent a single linguistic variable. Thus, fuzzification is the process of transforming a crisp value to a fuzzy value by identifying the quantity that is not fixed and uncertain. A membership function can be used to express this uncertainty.

Table 1 Description of various studies on Peer-to-Peer Network

Literature	Brief Description	Prime Consideration	Limitations
[50]	A survey on security issues that occur in the underlying P2P routing protocols, as well as fairness and trust issues that occur in file sharing and other P2P applications.	Trust in P2P system, Structured P2P Overlay Protocols, Routing and Security in P2P System	Limited discussion about unstructured overlays.
[51]	Framework is discussed for analyzing content distribution in peer to peer network.	Content Distribution, Security, Scalability, Performance and Fairness	Only non functional characteristics are focused
[52–55]	Peer selection and chunk selection mechanism	Priority based and slack time based scheduling scheme	Overall complexity is high
[56–58]	Hybrid overlay considering various components like Bandwidth, delay resistance etc.	Structured Overlay, Fuzzy logic based hybrid overlay	Space complexity is high
[59]	Identifying the various challenges in P2P file sharing system	Structured Overlay, Security, Routing, Free Riders	No discussion about the application based domain
[60–62]	Trustworthiness of the peers on past interactions and recommendations	Trust Management Between Peers	Security is the prime consideration only
[63]	Provided a new dataset for peer to peer live streaming applications	Peer-to-peer and CDN video streaming applications,	Doesn't Consider free riders and flash crowd
[64, 65]	Reviews and compares the P2P overlay network schemes and Categorizing them into groups of “structured” and “unstructured”.	Overlay, Structured and Un-structured	Only overlay is considered
[46]	Bandwidth-aware peer selection for P2P live streaming systems under	Bandwidth aware peer selection	No discussion about chunk selection
[9, 10, 12, 19, 20, 22, 50]	Flash crowd identification and admission control	Flash Crowd	Blocking based admission control
[66]	Distance driven based free rider problem solution	Live Video Streaming, Free Riders,	Lack of motivation for Free rides

Table 2 Comparison of Fuzzy Logic System

S. No	Fuzzy Logic System	Other Approaches
1	clarity and simplicity	Ambiguous and Hard to understand
2	Suitable for the Networking Environment	Difficult for a networking environment
3	Good and fast for approximation output	Good for exact output but slow
4	Robust in nature	Not robust
5	Modification is easy	Modification is complicated
6	Efficient solution for complex problems	Complex problem solutions are difficult
7	Flexible and easy to implement	Difficult to implement
8	Use Degree of membership	Strict on Boundaries

The outputs are also fuzzy values which are further processed by the inference engine in conjunction with the rule base to determine the specific output [58]. Here, input crisp values are remain bandwidth, pending request/buffer status and event-time. Equations 1, 2 and 3 describe the linguistic variables that are used in the technique. The linguistic variables chosen in this manner are more significant.

$$\text{Remain Bandwidth (R.B)} = \{\text{Low, Average, High}\} \quad (1)$$

$$\text{Pending Request (P.R)} = \{\text{Full, Medium, Empty}\} \quad (2)$$

$$\text{Event – Time (E.T)} = \{\text{Peak, Moderate, Free}\} \quad (3)$$

The proposed method makes use of the Mamdani inference engine, which fuzzifies all of the inputs and rules [67]. The trapezoidal membership function is utilized because it is a simple solution for a problem with three parameters [68]. The membership functions for the input variables bandwidth, buffer status, and event-time are shown in the Figs. 3, 4 and 5.

Figure 3 depicts the fuzzy membership function for remain bandwidth (R.B), which converts linguistic values of the peer's remain bandwidth into fuzzy values such as *Low*, *Average*, and *High*.

Figure 4 depicts the fuzzy membership function for pending request (P.R) which converts linguistic values of pending request of the peer into fuzzy values such as *Full*, *Medium*, and *Empty*.

Figure 5 depicts the fuzzy membership function for event-time which converts linguistic values of priority of the event into fuzzy values such as *Peak*, *Moderate*, and *Free*.

3.2 Inference rule for output

This section summarises all the rules which are included in the proposed method. Instead of using simply Boolean logic, a fuzzy inference system combines rules and a fuzzy membership function to handle the uncertainty of inputs Fig. 6. The rule establishes the relationship between the various linguistic input and output variables [58].

For example, if the value of remaining bandwidth is high, pending requests are low, and event-time is free, the output should be extremely low, as per the first rule in the given Table 3. The extremely low output means the chances of congestion in the network are also minimal. Table 3 discusses the details of the rule used in this approach. The total rules for all three linguistic variables, such as remaining bandwidth, pending request, and event

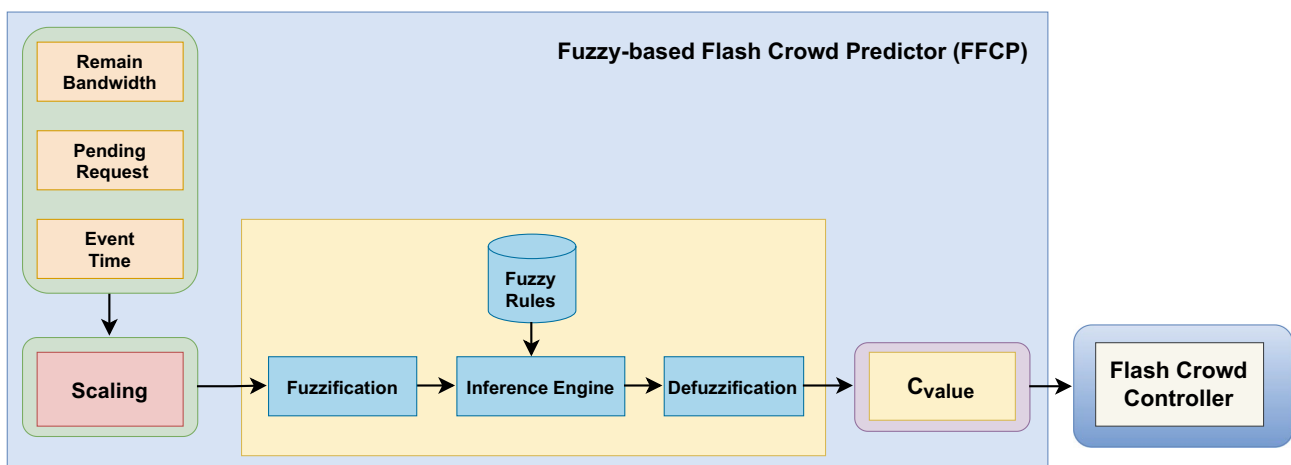
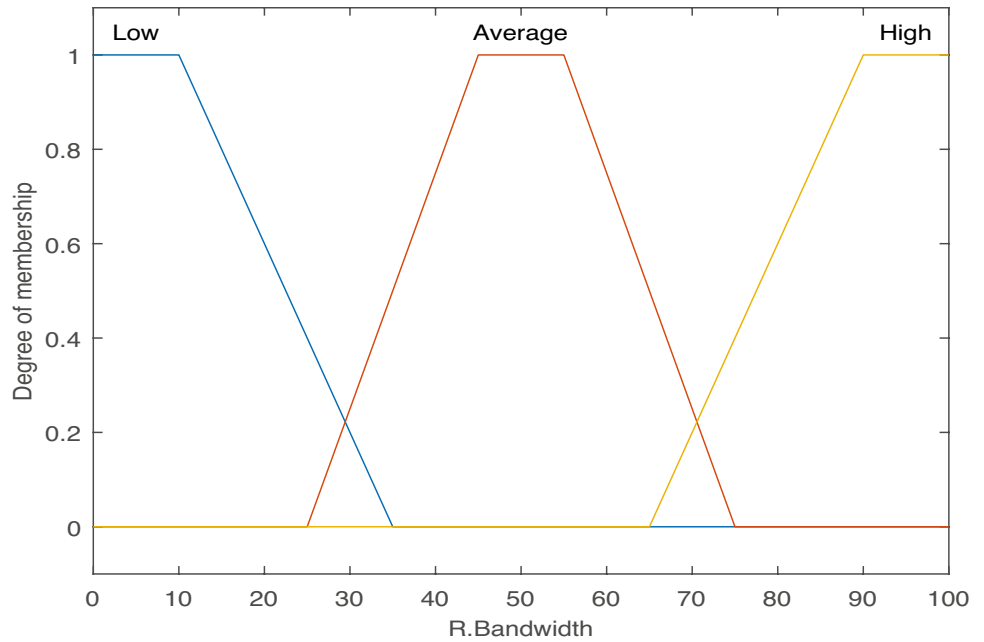
**Fig. 2** Fuzzy-based Flash Crowd Predictor

Fig. 3 Membership Function for Remain Bandwidth



time, are $3 \times 3 \times 3 = 27$. As a result, there are 27 priority values, which are distributed among six linguistic variables as given in Eq. 4.

$$Output = \begin{pmatrix} \text{Extremely - High} \\ \text{Very - High} \\ \text{High} \\ \text{Low} \\ \text{Very - Low} \\ \text{Extremely - Low} \end{pmatrix} = \begin{pmatrix} EH \\ VH \\ H \\ L \\ VL \\ EL \end{pmatrix} \quad (4)$$

3.3 Inference engine

The fuzzy inference engine uses logical reasoning in conjunction with empirical rules described as a rule base to derive the conclusion [58]. Fuzzy logic can manage network conditions like uncertainty and conflicting situations by applying membership functions [68]. The IF premise, THEN conclusion method is the most reliable approach to represent knowledge in fuzzy logic. The conclusion is the fuzzy output variable, while the premise is a set of

Fig. 4 Membership Function for Pending Request

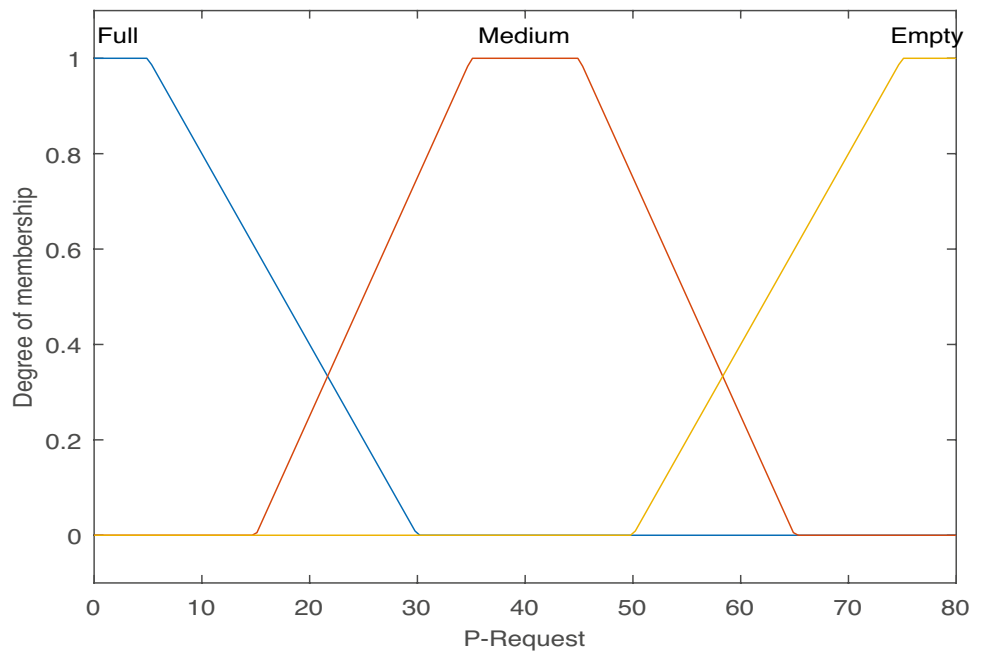
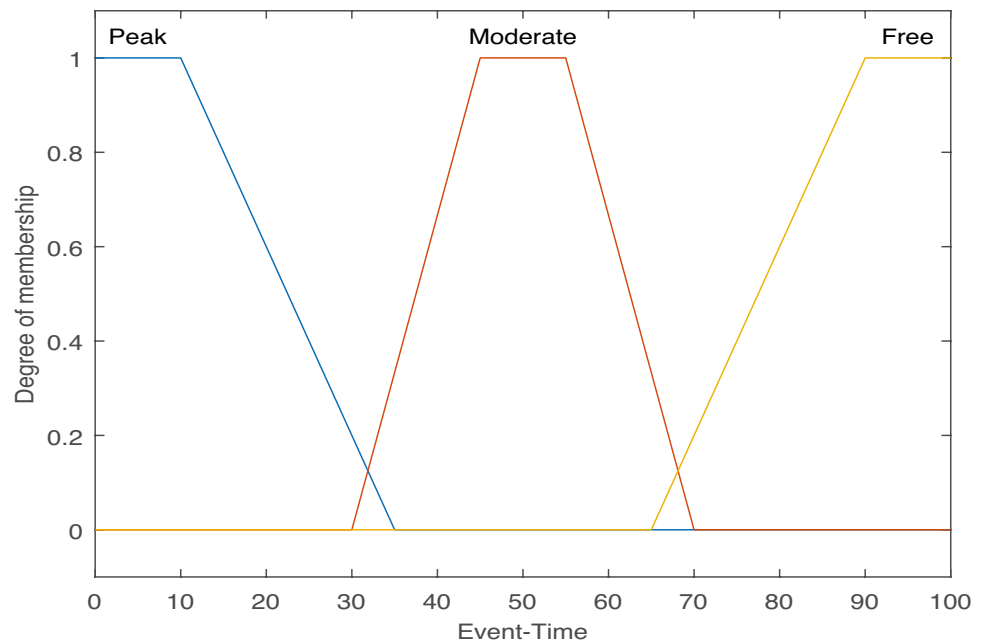


Fig. 5 Membership Function for Event Time



fuzzy inputs related to membership functions. IF-THEN rules are based on natural language representation of fuzzy logic sets. The linguistic variable changes depending on the current value of remain bandwidth, pending request, and event-time, and thus the output of the inference engine varies. The output in this case is desired in the form of network congestion. As shown in Table 3, the rules are designed, and a rule base is created.

3.4 Defuzzification

Defuzzification is the process of determining the crisp logic output from given fuzzy sets and membership functions. For defuzzification, the Mamdani centroid technique is used [58]. Based on remain bandwidth, pending request and event-time method, the best results for identification of congestion in the network are found. The center of

Fig. 6 Membership Function for Output

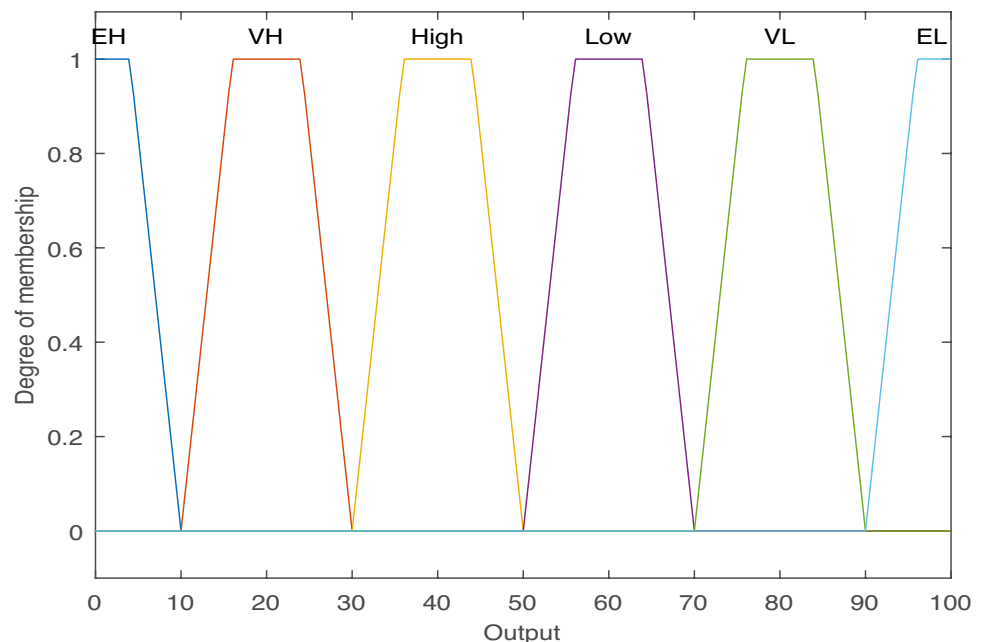


Table 3 Inference Rules

S.No	Rule
1.	If $R.B$ is High $\wedge$ $P.R$ is Less $\wedge$ $E.T$ is Free then <i>Output</i> is EL
2.	If $R.B$ is High $\wedge$ $P.R$ is Less $\wedge$ $E.T$ is Moderate then <i>Output</i> is EL
3.	If $R.B$ is High $\wedge$ $P.R$ is Less $\wedge$ $E.T$ is Peak then <i>Output</i> is VL
4.	If $R.B$ is High $\wedge$ $P.R$ is Medium $\wedge$ $E.T$ is Free then <i>Output</i> is EL
5.	If $R.B$ is High $\wedge$ $P.R$ is Medium $\wedge$ $E.T$ is Moderate then <i>Output</i> is VL
6.	If $R.B$ is High $\wedge$ $P.R$ is Medium $\wedge$ $E.T$ is Peak then <i>Output</i> is VH
7.	If $R.B$ is High $\wedge$ $P.R$ is High $\wedge$ $E.T$ is Free then <i>Output</i> is L
8.	If $R.B$ is High $\wedge$ $P.R$ is High $\wedge$ $E.T$ is Moderate then <i>Output</i> is L
9.	If $R.B$ is High $\wedge$ $P.R$ is High $\wedge$ $E.T$ is Peak then <i>Output</i> is EH
10.	If $R.B$ is Average $\wedge$ $P.R$ is Less $\wedge$ $E.T$ is Free then <i>Output</i> is EL
11.	If $R.B$ is Average $\wedge$ $P.R$ is Less $\wedge$ $E.T$ is Moderate then <i>Output</i> is L
12.	If $R.B$ is Average $\wedge$ $P.R$ is Less $\wedge$ $E.T$ is Peak then <i>Output</i> is VH
13.	If $R.B$ is Average $\wedge$ $P.R$ is Medium $\wedge$ $E.T$ is Free then <i>Output</i> is VL
14.	If $R.B$ is Average $\wedge$ $P.R$ is Medium $\wedge$ $E.T$ is Moderate then <i>Output</i> is L
15.	If $R.B$ is Average $\wedge$ $P.R$ is Medium $\wedge$ $E.T$ is Peak then <i>Output</i> is VH
16.	If $R.B$ is Average $\wedge$ $P.R$ is High $\wedge$ $E.T$ is Free then <i>Output</i> is L
17.	If $R.B$ is Average $\wedge$ $P.R$ is High $\wedge$ $E.T$ is Moderate then <i>Output</i> is L
18.	If $R.B$ is Average $\wedge$ $P.R$ is High $\wedge$ $E.T$ is Peak then <i>Output</i> is EH
19.	If $R.B$ is Low $\wedge$ $P.R$ is Less $\wedge$ $E.T$ is Free then <i>Output</i> is H
20.	If $R.B$ is Low $\wedge$ $P.R$ is Less $\wedge$ $E.T$ is Moderate then <i>Output</i> is VH
21.	If $R.B$ is Low $\wedge$ $P.R$ is Less $\wedge$ $E.T$ is Peak then <i>Output</i> is VH
22.	If $R.B$ is Low $\wedge$ $P.R$ is Medium $\wedge$ $E.T$ is Free then <i>Output</i> is H
23.	If $R.B$ is Low $\wedge$ $P.R$ is Medium $\wedge$ $E.T$ is Moderate then <i>Output</i> is VH
24.	If $R.B$ is Low $\wedge$ $P.R$ is Medium $\wedge$ $E.T$ is Peak then <i>Output</i> is EH
25.	If $R.B$ is Low $\wedge$ $P.R$ is High $\wedge$ $E.T$ is Free then <i>Output</i> is H
26.	If $R.B$ is Low $\wedge$ $P.R$ is High $\wedge$ $E.T$ is Moderate then <i>Output</i> is VH
27.	If $R.B$ is Low $\wedge$ $P.R$ is High $\wedge$ $E.T$ is Peak then <i>Output</i> is EH

gravity method, developed by [69], is used to determine the level of congestion upon the each peer's entry. The Mamdani implication criteria are mapped into linguistic values that describe the level of inference activity.

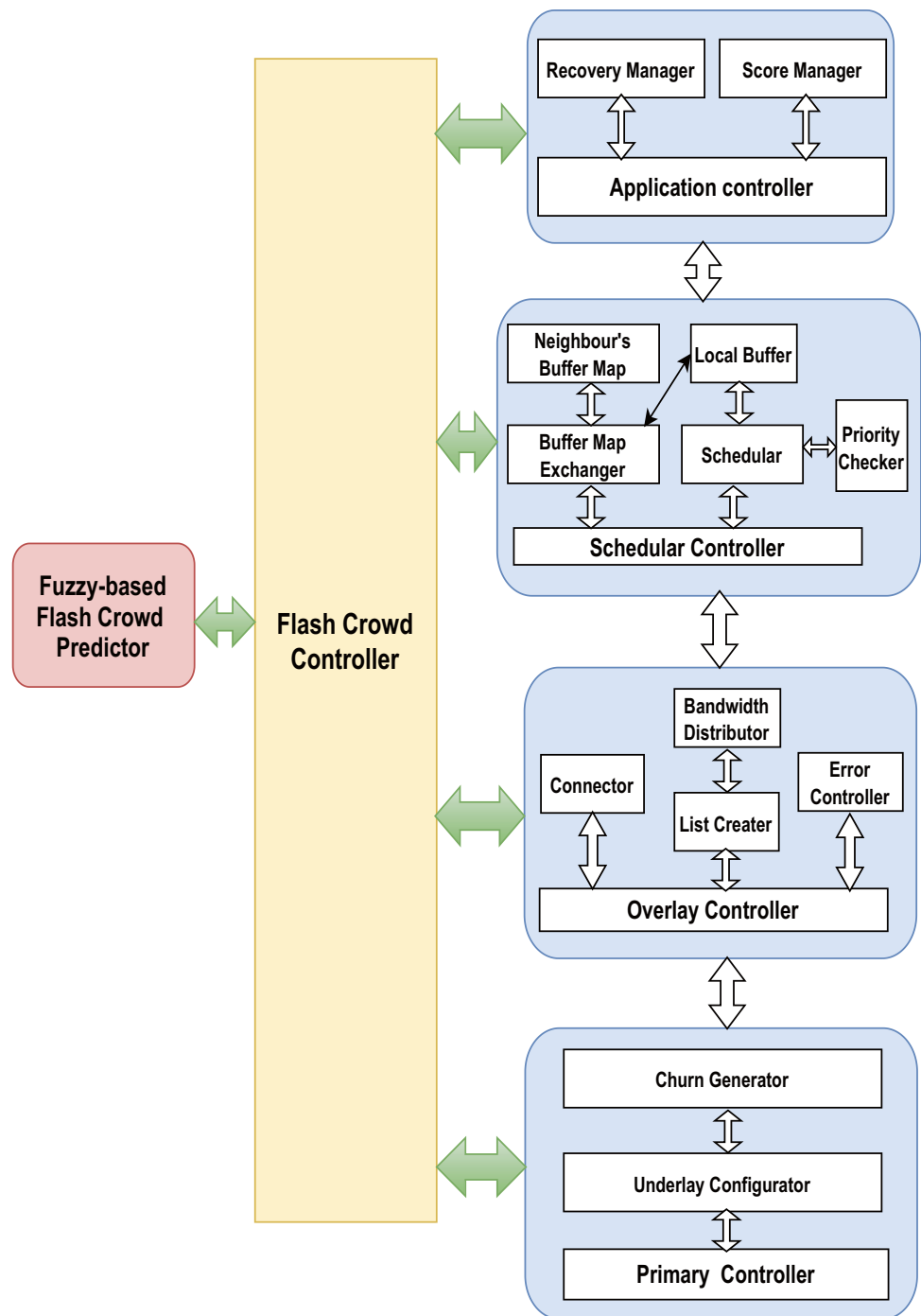
4 Fuzzy based hierarchical flash crowd controller (FHFC²)

This section presents the proposed Fuzzy based Hierarchical Flash Crowd Controller (FHFC²) for a P2P live video streaming system. The existing approaches discussed in section 2 perform admission control at the application level and rely on restricting entry and/or blocking the peers after a specific measure. In addition, these approaches do not consider other crucial factors such as remain bandwidth, peak time, pending request and event priorities for handling the flash crowd. These shortcomings of the existing schemes for flash crowd management motivated us to investigate this problem and develop an enhanced solution. To the best of our knowledge, no researchers have tried the application of fuzzy logic in

designing the solution for flash crowd management in the P2P networks. In this work, we have developed a fuzzy-based flash crowd predictor (FFCP) as discussed in the previous section. The prediction of flash crowd in advance is essential for any system to take timely corrective measures and avoid the degradation in this service to users. The proposed approach is designed and developed to provide a hierarchical-based solution for flash crowd management. Unlike the simple admission control strategy the proposed approach explores the various options before denying the request of a peer when the system is overloaded. These options are creating new overlays, and use an adaptive scheduling scheme motivating existing peers to upload the contents thereby catering for the need of other peers in the system. Finally, the admission control strategy is used. The core concept of the proposed approach is to provide a fuzzy-based hierarchical flash crowd controller (FHFC²) to handle the flash crowd efficiently and the framework for the same is depicted in Fig. 7.

Algorithm 1 is used to process the request of a new peer entry in the network. The working of the proposed algorithm is explained. A new peer i sends the neighbour request $NREQ_i^T <$

Fig. 7 Fuzzy based Hierarchical Flash Crowd Controller



B_{TU}^i, C_{id}, G_i to tracker T for joining the network. The request contain multiple variables such as peer bandwidth (B_{TU}^i), video content id (C_{id}) and geographical location (G_i). Content id (C_{id}) is the identification number of the video that the peer i requested for. Once the neighbour request is received, tracker

T will check the status of the peer i . The tracker will identify the status of the peer i whether it's a new peer or a old peer requesting for new session. Old peer will assign the parameter values (utilization, age) from the database maintained by the tracker and the new peer is assigned default values.

Algorithm 1 $FHFC^2$ (B_{TU}^i, C_{id}, G_i)

Input: (B_{TU}^i, C_{id}, G_i)
Output: Smooth Video Playback

```

1:  $NREQ_i^T < B_{TU}^i, C_{id}, G_i >$   $\triangleright$  Request from client  $i$  to Tracker  $T$ 
2: Fuzzy logic based Traffic Prediction()
3: IF (Prediction == 1)
4:   Congestion Control Mechanism()
5: else
6:   Det_Neighbour(D)
7:    $RES_T^i(L_i)$   $\triangleright$  Response from Tracker  $T$  to client  $i$ 
8:    $C_{req}_i^j$  where ( $j \subseteq L_i$ )  $\triangleright$  Connection request from client  $i$  to
     peer/server*  $j$ 
9:    $C_{res}_j^i$  (Positive/Negative)  $\triangleright$  Connection response from peer/server*
      $j$  to client  $i$ 
10:  if( $C_{res}_j^i$ ) goto 11 else 8 for new  $j$  from  $L_i$ 
11:  Send  $C_{req}_P^i$   $\triangleright$  where  $P \subseteq L_i$ 
12:  According to RTT time finds the best response
13:  If continue step (11) else leave the network.
14:  Call get chunk( $C_{req}_P^i$ )
15:  Periodically update status of Tracker.
```

After receiving request from the peer i , tracker T predicts the flash crowd in the network using the proposed scheme fuzzy-based hierarchical flash crowd controller ($FHFC^2$). The Fuzzy logic based flash crowd prediction (FFCP) model is invoked as discussed in Algorithm 2. Tracker T calculates all the crucial parameters like buffer size status of each peer/server available in the network, updates the remain bandwidth, peak time, pending request and event priorities. Then the tracker applies the fuzzification (Algorithm 2 line 7) and fuzzy rules using these crucial parameters. According to the fuzzification, the prediction value is evaluated. If the predicted value is 1, it indicates the congestion scenario in the network and refers to flash crowd controller to take the appropriate congestion control scheme (Algorithm 1 line 4).

Algorithm 2 FFCP (B_{TU}^i , Buffer Size, $Event$, $Time$)

Input: (B_{TU}^i , Buffer Size, $Event$, $Time$)
Output: Prediction Value and Congestion Level

```

1: For each peer  $i$ 
2:   Find Buffer Size of all connected Peers.
3:   Find the Buffer Size  $B_{si}$ 
4:   Find Remain Bandwidth  $\sum B_i$ 
5:   Find the Event-Time  $ET_i$ 
6:   Fuzzification ( $B_{si}, \sum B_i, ET_i$ )
7:   Apply Fuzzy Rules
8:   Defuzzification and Evaluate Prediction Value
9:   Return Prediction Value and Congestion Value ( $C_{Value}$ )
10: End For
```

Flash Crowd Controller is the heart of Fuzzy-based Hierarchical Flash Crowd Controller ($FHFC^2$). Flash Crowd Controller follows a hierarchy-based congestion control mechanism (CCM) for controlling the congestion in the network. According to the congestion value C_{Value} predicted by FFCP,

tracker finds the severity of the congestion in the network. If C_{Value} is less than Th_1 (Primary Threshold value), it is considered to be an ideal network. In this scenario, Flash Crowd Controller will allow the peer to continue in the network without any constraint. If C_{Value} is in between Th_1 and Th_2 then, it shows the initial congestion. In this condition, network congestion due to flash crowd is under control, and the network can accept more peers with restricted entries. The controller will set the primary mode in this scenario. When the primary mode is on the tracker will check for each request and allow only good peers in the network. Controller also sends the status to all the sub-controllers. Good peers are those peers which consume less bandwidth as compared to other peers. According to the past history and present requests, only good peers participate in the network so that the requirement of each and every available peer can be fulfilled without any issue.

If C_{Value} is in between the Th_2 and Th_3 , the network is considered more congested as compared to the primary mode and this is considered a symptom of mild congestion and flash crowd controller invokes the overlay controller. Overlay controller starts the creation of a new overlay over the existing one. The newly created overlay is named as Bandwidth-Utilization Aware Overlay as discussed in Algorithm 3. Overlay controller finds the list of best available peers in the network and these peers are given priority on the newly created overlay.

Algorithm 3 Congestion Control Mechanism (CCM (C_{value}))

Input: C_{value}
Output: Different Mode

```

1: if( $C_{value} < Th_1$ )
2:   Ideal Mode
3: else if ( $Th_1 < C_{value} < Th_2$ )
4:   Primary Mode()
5: else if ( $Th_2 < C_{value} < Th_3$ )
6:   Overlay Controller()
7: else ( $Th_3 < C_{value} < Th_4$ )
8:   Scheduler Mode
9:   For each new request  $NREQ_i^T < B_{TU}^i, C_{id}, G_i >$ 
10:    Quality Aware Scheduling Scheme()
11:    Send the motivation packets
12:  end for
13: else
14:  Application Mode()
```

Algorithm 4 Primary Mode

Input: New Request(s)
Output: Supply the requirement

```

1: Inform all Controller
2: Req-Band  $\leftarrow$  Demand. $NREQ_1^T$ 
3: For each new requests  $NREQ_i^T < B_{TU}^i, C_{id}, G_i >$ 
4:   if Demand. $NREQ_{i+1}^T < Req-Band$ 
5:     Req-Band  $\leftarrow$  Demand. $NREQ_{i+1}^T$ 
6:   end if
7: end for
```

Algorithm 5 Overlay Mode

Input: New Request(s)
Output: Supply the requirement

- 1: For each new request $NREQ_i^T < B_{TU}^i, C_{id}, G_i >$
- 2: Bandwidth-Utilization Aware Overlay Approach
- 3: Send the motivation packets
- 4: end for

Algorithm 6 Bandwidth-Utilization Aware Overlay Approach

Input: Available Peers (D)
Output: Best Available Peers List (L_i)

- 1: if ($L_S < \text{Max}(S)$)
- 2: $L_i.ADD(S)$
- 3: level $\leftarrow 1$
- 4: While(level $< L_vel_i$ && $L_i < Th_5$)
- 5: { For(each peer at the level i)
- 6: { If($UT_i > Th_6$ && and list id not full)
- 7: Add(P_i) in the L_i }
- 8: If(L_i have space)
- 9: { For(each peer at the level i)
- 10: { If($UT_i > Th_5$ && P_i list in not full && P_i contains lower level
- 11: peer)
- 12: ADD(P_i) in List }
- 13: Swapping }
- 14: increase level; }
- 15: restart level from 1
- 16: while(level $< L_vel_i$ && $L_i < Th_6$)
- 17: { For(each peer at the level i)
- 18: { If(List of P_i is not full)
- 19: Add(P_i^i) in List }
- 20: If($L_i < Th_6$)
- 21: { For(each peer at the level i)
- 22: { If(P_i contains lower level peer L_vel_i && P_i is not full)
- 23: Add(P_i) in the List }
- 24: Swapping }
- 25: level ++;

The detailed information of all available peers in the network is given as the input to Algorithm 3 and the output provides the list of best available peers (L_i) determined based on their upload bandwidth and utilization. Flash crowd controller first checks whether the server can connect more peers or not. If the server have sufficient bandwidth to connect with more peers the server is added to list (L_i) of the client i as the server at the highest level (level 0). Then tracker checks all the available peers in the network up to the level(L_vel_i) or until the list (L_i) is not full. At each level, peer j {where $j \in \text{Database}(D)$ } is inserted into the list (L_i). If j is a more utilized peer and client have at least one space left or peer j is full but connected to a peer k whose level is less than L_vel_i , the swapping takes place. At each level, same procedure is repeated until the list is full or the level is complete. After traversing up to the level (L_vel_i), if still the list (L_i) have more places then other peers are inserted according to upload bandwidth and repeat the same procedure for each level from 1 to (L_vel_i).

After creating the list (L_i), the tracker sends a response (RES_T^i (L_i)) to client i (Algorithm 1 line number 7) that contains the list of best available peers for i . After receiving the neighbours' list (L_i) from the tracker, client i send the connection request

($C_req_i^j$) to peers j from the list and connections are established. Periodically peers update their status to the tracker.

If C_{value} is in between the Th_3 and Th_4 , it represents that the congestion level in the network is more severe as compared to the other two cases. So flash crowd controller informs the overlay controller as well as the scheduler. The overlay controller and scheduler start the admission control mechanism. Overlay controller finds the list of best available peers in the network and further, these peers have given priority for overlay creation Algorithm 3. The scheduler controller also informs the scheduler to initiate the congestion control. Scheduler adopted the 'Quality Aware Scheduling Scheme' to fulfil the requirement of newly joined peers. The flash crowd controller also sends the motivation packets to the peer available in the network. Motivation packets inform the peers' that they need to participate in the network positively by uploading content in the network. Selfless peers are self-motivated. So, motivation packets do not affect the performance of these peer but selfish peer need to participate positively in the network otherwise their service will be discontinued and they will not be allowed to utilize the resources of the network. 'Quality Aware Scheduling Scheme' is demonstrated in Algorithm 5. In Algorithm 5, the connection request $C_req_i^T$ is sent from client i to parent peers P . Parent peers exchange their buffer maps with client i . The buffer map (B_f) is periodically exchanged between the peers to inform about the status of their buffer and ascertain the requested chunk.

Algorithm 7 Quality Aware Scheduling Scheme()

Input: ($C_req_i^P$)
Output: (C_k, P)

- 1: $C_req_i^j$ Receive by peer j
- 2: Exchange Buffer map (Bf)
- 3: Calculates $RTT_{ij} \forall x : x \in P_i$
- 4: Call Startup_buffer_loc(S_0, S_P, S_i)
- 5: Check for buffer map periodically
- 6: if(layer==Base && Scheduler Signal ==1)
- 7: Push phase
- 8: else
- 9: Call Chunk Selection Procedure(Bf_j)

Client i calculates the RTT (Round Trip Time) for the reply of each parent and according to the RTT value clients' peer finds the best peer among the parent peers. The other peers are considered backup parent peers and are used according to their requirements. The use of RTT is useful for calculating the real-time condition of the network. A peer may be statistically best, but dynamically it is possible that the network is congested or RTT is very high due to other reasons like geographical distance. The Startup_buffer_loc() algorithm is used to find an appropriate chunk after buffer map exchange between the peers [39, 55]. This ensures the timely delivery of required chunks and the user can enjoy the smooth video at the receiver end.

After getting the appropriate start-up location, the sender peer initiates the push phase and sends the available base layer data to the requesting peer. The push phase is used to fulfil the minimum requirements of each and every available peer in the network. The push phase is simple as the sender sends the data to each of its child peer without waiting for the response from the receiving child peers. As mentioned above, the sender pushes only the available base layer and if any layer chunk is missing or enhanced layer chunk is required or the congestion value set by the scheduler is 0, the chunk selection procedure is the same as discussed in [55].

Algorithm 8 Application Mode

Input:New Request(s)
Output:Supply the requirement

- 1: For each new request
- 2: Drop All new Request
- 3: Create waiting List WL_i for future
- 4: Start Recovery Mode
- 5: For each peer i
- 6: find the victim peer
- 7: Blocked from the Network
- 8: Repeat until the network is in safe mode
- 9: end for
- 10: end for

The C_{Value} greater than Th_4 represents the extreme level of congestion. So congestion controller informs all the sub-controller in the network about the status of congestion. All the sub-controller starts the congestion control mechanism. The overlay controller and scheduler controller follow the same procedure as discussed in the Algorithms 3 and 5, respectively.

The application controller restricts the immediate entry of newly requested peers i in the network and insert them in a waiting list W_i . The waited peers request may be entertained if the system came into a minimum congestion state i.e. normal state. The application controller also starts recovery mode as discussed in Algorithm 7.

If the congestion in the network still persists after restricting the entry of new peer as discussed above, the controller needs to enter into the recovery mode. The algorithm used recovery mode as given in Algorithm 7. In recovery mode, it is necessary to bring the congestion in the network to acceptable limits and therefore, some of the peers are to be removed from the existing network. Algorithm 7 selects such peers and can be referred to as victim peers. The victim peers are selected based on the score available with each peer. This score is calculated based on the reputation and the behaviour of the peer as discussed in [39]. The peers having low score are selected as a victim and removed sequentially from the network till the situation become acceptable. The above discussed strategy used by the proposed controller is repeated whenever the network becomes overloaded. The proposed approach is implemented, its criticality and performance is discussed in the subsequent sections.

Table 4 Nomenclature

Notation	Definition
TBA^i	Total Bandwidth Available of peer i
TBR_L^i	Total Bandwidth Required for peer i Upto layer L
$UB_{Active-Peer}$	Upload Bandwidth of Active Peer
R_i^j	Request i for peer j
N	Total Number of Peers in the Network
UBA_i	Upload Bandwidth Available for Peer i
UBR_i	Upload Bandwidth Required for Peer i
TDB^i	Total Downloaded Bandwidth of peer i
TUB^i	Total Uploaded Bandwidth of peer i
λ	Number of motivated parents peers
$ChunkSizeof(Req_i^j)$	Chunk size of request peer i Upto Layer L
TB_{Free}	Total Bandwidth of Free Peer

5 Criticality of the algorithm

Corollary 1 *Trivial Condition: A condition for Minimum requirement of bandwidth for a peer i to be enjoy un-interrupted video quality. $\sum_{j=1}^x UB_{Active-Peer} > TBR$ and $TBA > \sum_{j=1}^x UB_{Active-Peer}$*

The nomenclature used in the critical assessment of the algorithm is given in Table 4.

Lemma 1 *In primary mode, any set of peers $P_j \in P$, Where $P \in A \cup B$ is sufficient to fulfil the requirement of all the possible request R_i^j for peer j ($TBA > \sum_{i=1}^k \sum_{j=1}^x R_i^j$).*

Proof: For the proof, we consider that each request R_i^j for peer j should be entered in the queue. We use the induction on the number of queue operations. Initially when the Q contains only the first request R_1^1 so lemma certainly holds due to $TBA = \text{Server Capacity}$.

Now for the inductive step, we assume that the inductive hypothesis holds up to n number of peers, so Eq. 5 will hold.

$$TBA > \sum_{i=1}^k \sum_{j=1}^N R_i^j \quad (5)$$

where $N = x + y$, Now lets consider new peer p joins the network and for $N + 1$ peer, when p join the network there will be two cases ($x = x + 1$ and $y = y$) or ($x = x$ and $y = y + 1$). x and y are active and passive peers respectively. **Case 1:** New entered peer p is an active peer so equation so 6 will hold

$$UBA_p > UBR_p \quad (6)$$

From corollary 1 Eqs. 7 and 8 are true.

$$\sum_{j=1}^x UB_{ActivePeer} > TBR^x \quad (7)$$

$$TBA^x > \sum_{j=1}^x UB_{ActivePeer} \quad (8)$$

$$\sum_{j=1}^x UB_{ActivePeer} + UBA_p > TBR^x + UBR_p \quad (9)$$

$$\sum_{j=1}^{x+1} UB_{ActivePeer} > TBR^{x+1} \quad (10)$$

so, for $N=N+1$ inequality Eq. 5 holds.

Case 2: New entered peer p is a passive peer then $x = x$ and $y = y + 1$. Algorithm 4 will make sure the inequality Eq. 5. So, inequality will hold for both the cases.

Lemma 2 In the network, when a new overlay construction approach will be applied and a new selection of parents' peer is applicable. The overlay manager will motivate the peers.

Proof: For the proof, let's assume the parents' peer j is which is getting demotivated and its upload bandwidth is less as compared to downloaded bandwidth. If this is the case then $TDB_i > TUB_i$. For participating in the network peers' will maintain Eq. 6. If the peer is not participating then there may be two possible cases.

Case 1: Peer i is not connected in this case peer will be removed from the network.

Case 2: Peer i is intentionally not participated in the network. Then peer will be considered as a passive peer Lemma 1 case 2.

Lemma 3 For handling the mild congestion, the overlay manager has sufficient resources.

Proof: If the congestion in the network is ideal then lemma 2 will make sure the quality of the video. If there is more congestion and a mild congestion level is hit, the quality manager will make sure to provide the required quality.

The overlay manager improves the selection of parents' peers and motivates the peers to share more bandwidth. Let's assume λ is the number of parent peers which are motivated by the overlay manager.

$$UB_{ParentPeer} = \sum_{j=1}^{x+\lambda} UB_{ActivePeer} + \sum_{j=1}^{y-\lambda} UB_{PassivePeer} + \sum_{j=1}^{\lambda} UB_{Motivated-Bandwidth} \quad (11)$$

where λ can't be negative (lemma 7.2). For all positive values of $\lambda = 1$ inequality Eqs. 7 and 8 will be true.

Lemma 4 When the network suffers from mild congestion, the overlay manager will help to achieve the desired quality in the network.

Proof: If flash crowd symbols are generated by the network and there is more number of requests than for each request check the utilization of each peer. If lemma 2 fails and the threshold lie between 2 and 3 then the overlay controller will create parents list using available bandwidth and utilization of past and present sessions. Initially

$$\sum_{j=1}^x UB_{Active-Peer} > TBR \quad (12)$$

Bandwidth and utilization-based overlay approach improve the overall interested bandwidth in the network. As the number of interested peer increases, that helps to find the updated parent peers. Updated parents' peers are increased and λ peers are interested peers. λ is the number of new peers which are having higher bandwidth and utilization factors.

$$UB_{Parents} = \sum_{j=1}^{X+t+\lambda} UB_{Active-peers} + \sum_{j=1}^{Y+t-\lambda} UB_{Passive-peers} + Motivation Bandwidth \quad (13)$$

so, the updated list of parents' peer improves the overall available bandwidth and further help to improve the peers in the network to enjoy better video quality.

Lemma 5 Scheduler will provide the chunks before its deadline to maintain the minimized required bandwidth in the network.

Proof: When the threshold is between th_2 and th_3 . The scheduler will maintain the required quality of service by applying the fundamental essentials for each peer. Schedulers apply the quality aware scheduling scheme (Algorithm 7) and restricted the demand of higher quality video to the peers in the network and just supply the basic requirement in the network. Applying the quality-aware adaptive scheduling scheme will effect the TBR as shown in Eq. 14. Where

$ChunkSizeof(Req_i)$ is calculated by Eq. 15, N is the total number of servers and peers in the network and M is total requests sent by the nodes N . Updated $ChunkSizeof(Req_i)$ is calculated as shown in Eq. 16.

$$TBR = \sum_{j=1}^N \sum_{i=1}^M P_j * UChunkSizeof(Req_i) \quad (14)$$

$$ChunkSizeof(Req_i) = \sum_{i=1}^l ChunkSizeof(Req_i^l) \quad (15)$$

$$UChunkSizeof(Req_i) = \sum_{j=1}^l ChunkSizeof(Req_i^j) - \sum_{j=p}^l ChunkSizeof(Req_i^j) \quad (16)$$

where L is the updated layer number up to which the scheduler can handle without entering in the flag value above threshold 3. So, the updated value of TBR_L is as shown in Eq. 17.

$$TBR_L = \sum_{j=1}^N \sum_{i=1}^M P_j * UChunkSizeof(Req_i) \quad (17)$$

so,

$$TBR = TBR_L + \sum_{j=1}^N \sum_{i=1}^M \sum_{l=L}^l P_j * ChunkSizeof(Req_i^l) \quad (18)$$

because

$$\sum_{j=1}^N \sum_{i=1}^M \sum_{l=L}^l P_j * ChunkSizeof(Req_i^l) > 0 \quad (19)$$

so, by Eqs. 18 and 20

$$TBR > TBR_L \quad (20)$$

now applying the transitivity property on the inequality of Eqs. 20 and 7 we get Eq. 21.

$$\sum_{j=1}^X UB_{Active-Peer} > TBR > TBR_L \quad (21)$$

that further improves the overall experience on the receiver side.

Lemma 6 If the flag value is more than the maximum flag value, removal of victim peers leads to providing sufficient requirements of demands.

Proof: When threshold shows that there is extreme congestion in the network due to flash crowd. The application controller will take care of Eq. 22. The controller starts dropping all the new request and start creating a waiting list for the future, and also start the recovery mode. In the recovery mode, the controller find the victim peers' and remove from the system. For communicating in the network in ideal conditions, the system must follow $\sum_{j=1}^X UB_{Active-Peer} > TBR$ and $TBA > \sum_{j=1}^X UB_{Active-Peer}$. But as the level of flash crowd increase the condition is no more sustain. Recovery mode increases the Free bandwidth as evaluated in Eq. 20.

$$TB_{Free} = \sum_{j=1}^v UB_j \quad (22)$$

where v is the number of victim peers. Identification of victim peers is done using score-based incentive mechanism [39] and removed from the network. Peers are removed from the network until the network comes again in safe mode.

So, updated TBR_u is as follows in Eq. 23

$$TBR_u = TBR - TB_{Free} \quad (23)$$

Safe mode is the scenario where the trivia condition is fulfilled. It further leads to fulfill the trivia condition : $\sum_{j=1}^X UB_{Active-Peer} > TBR$.

Theorem 1 Required bandwidth for each peer i to enjoy uninterrupted quality of video after the deployment of fuzzy-based hierarchical flash crowd control mechanism in the P2P Network should be sufficiently large.

Proof: To proof the correctness of the Theorem we have to consider different scenarios.

1. If the scenario of Ideal network (Without Congestion) then lemma-1 will provide the solution of bandwidth problem.
2. If the network faces initial level congestion (which can be identified by the Th_1 and Th_2). The primary mode is activated which further restricted the entry of new peers into the network. Peers with better past records and minimized requirements are allowed to participate in the network. Lemma-2 proves that even after facing an initial level of congestion FHFC² provides smooth-quality video to each available peer in the network.
3. If the congestion level fall in the range of middle congestion and the flag value is set according to that. In the middle congestion scenario, the overlay manager makes sure that demand of each peer can be fulfilled by using bandwidth-utilization aware overlay approach.

Bandwidth-utilization-aware overlay approach make sure that the available bandwidth of the parent peer is sufficiently available. Also motivated peers also help to achieve the same. Lemma-3 proof that even after facing middle level of congestion HLFAC provide the smooth quality of video to each and every available peer in the network.

4. *If the congestion level fall in the range of serious level congestion and the flag value is set according to that. In a serious level congestion scenario, the scheduler manager makes sure that demand of each peer can be fulfilled by using quality-aware adaptive scheduling scheme. quality-aware adaptive scheduling scheme make sure that the minimum requirement of each available peer in the network is fulfilled by the scheduler (Lemma-5). Lemma-3 proves that even after facing a middle level of congestion FHFC² provides the smooth-quality of the video to each available peer in the network.*
5. *If the congestion level fall in the range of extreme-level congestion and the flag value is set according to that. In extreme level congestion scenario, the application manager restricts the new entry in the network. Further, drop all the new requests made by the peers and start recovery mode. In the recovery mode application manager will find the culprit peer using a score-based incentive mechanism. SIM approach find the selfish peers in the network which is utilizing the resources of the network without positive participation. Positive participation means sharing resources. The system will remain in recovery mode until the system come to a safe state. Safe state refers to the scenario of a trivial condition. Lemma-6 proves that even after facing an extreme level of congestion FHFC² provide the smooth-quality of the video to each available peer in the network.*

A peer-to-peer network can undergo different scenarios under flash crowd conditions as discussed above. Fuzzy-based Hierarchical Flash Crowd Controller (FHFC²) provide the solution for each scenario.

6 Results and discussion

The proposed approach is implemented using the OMNET++ simulation model in the OverSim framework [70]. The comparative analysis of the proposed FHFC² approach with respect to a distribution of playback delay, end-to-end delay, start-up delay, and total distortion is shown in Figs. 8, 9, 10, and 11. Table 5 demonstrates various parameters and their values with respect to load variations. Apart from the results obtained from our proposed scheme FHFC², we have simulated the scheme in four different ways for comparative analysis. The different schemes are - WAC: Without

Admission Control; PL (Primary Level): the scheduler mode is on after detection of primary congestion; SL (Scheduler Level): the scheduler mode is on after critical congestion; BJS (Batch Join scheme). BJS is a completely P2P live video streaming flash crowd reduction scheme.

6.1 Result analysis

From Table 5, it can be observed that when the network is not congested and the number of peers is 1000, various basic parameters of the peer-to-peer network and live video streaming remain under control. And when the number of peers is 1100, the overload in the network is 10%. From this, it could be inferred that when the overload increases then the performance of the network start decreasing, this could easily be observed from the respective table. Further, when the overhead is 10%, the FHFC² approach motivates 3% of existing peers while 3% of newly requested peers are restricted from participation in the network and added in WL_i . Without admission, the control network can handle approximately 7% overload and beyond this, the system goes to a halt and that is why it is not reported in Table 5. Moreover, when overload increases a significant difference can be observed from the Table 5, which results that when the overhead is 20%, the FHFC² approach also blocks the peer from participating in the network according to the score-based incentive approach, while the motivated peers are approximately 4% and restricted peers are 5% which helps to handle the overload and improves the quality of peer and performance of the network as compared to other approaches (WAC, PL, SL) of congestion control mechanism. As the overload further increase by 30%, 40% and 50%, the fuzzy-based hierarchical flash crowd controller mechanism helps to motivate the peer using score-based incentive mechanism. The FHFC² also blocked the peers who behaves selfish in the network. Blocked peer, motivated peers and restricted peers are (5%, 6% and 8%), (6%, 6.4% and 12%) and (8%, 7% and 15%) for 30%, 40% and 50% overload respectively. The PL variant of admission control can handle approximately 14% overhead, BJS can handle 16% overhead while the SL variant of the proposed controller can handle 25% overload in the network and that's why the subsequent values are not reflected in Table 5. BJS is not motivated by the peers so relevant values are not available in the table. The proposed method is able to handle more than 50% of overload convincingly.

Figure 8 is created for playback delay considering CDF and time in seconds. The result shows that 100% of peers in FHFC² have a playback delays of less than 42 seconds, 40% of peers have a playback delays of less than 32 seconds, and 80% of peers have a playback delay less than 25 seconds. From the figure, it can be easily observed that the proposed FHFC² approach outperforms the BJS approach as 40% of peers in BJS have a playback delay

Table 5 Results of proposed approach and other schemes

Flash Crowd Control	Peers Over-loaded	Playback Delay	End to End Delay	Total Distortion	Start-up Delay	Blocked Peers	Motivated Peers	Restricted Peers
<i>FHFC</i> ²	0%	23.49	12.29	20.0	51.89	0%	2%	2.5%
WAC	0%	21.92	11.56	18.62	48.92	—	—	—
PL	0%	22.18	12.12	19.75	50.48	—	—	—
SL	0%	23.12	12.10	19.56	50.31	—	—	—
BJS	0%	22.14	12.12	19.68	48.78	—	—	—
<i>FHFC</i> ²	10%	24.28	13.52	20.59	53.84	0%	3%	3%
WAC	10%	—	—	—	—	—	—	—
PL	10%	23.29	13.48	20.50	52.72	—	—	—
SL	10%	24.05	13.48	20.55	53.62	—	—	—
BJS	10%	23.50	13.52	20.45	54.68	—	—	—
<i>FHFC</i> ²	20%	25.16	14.15	21.52	55.24	2%	4%	5%
WAC	20%	—	—	—	—	—	—	—
PL	20%	26.89	15.19	22.50	54.31	—	—	—
SL	20%	—	—	—	—	—	—	—
BJS	20%	—	—	—	—	—	—	—
<i>FHFC</i> ²	30%	27.95	16.7	21.80	56.24	5%	6%	8%
WAC	30%	—	—	—	—	—	—	—
PL	30%	—	—	—	—	—	—	—
SL	30%	—	—	—	—	—	—	—
BJS	30%	—	—	—	—	—	—	—
<i>FHFC</i> ²	40%	30.21	19.92	22.00	57.30	6%	6.40%	12%
WAC	40%	—	—	—	—	—	—	—
PL	40%	—	—	—	—	—	—	—
SL	40%	—	—	—	—	—	—	—
BJS	40%	—	—	—	—	—	—	—
<i>FHFC</i> ²	50%	32.25	19.2	22.54	58.30	8%	7%	15%
WAC	50%	—	—	—	—	—	—	—
PL	50%	—	—	—	—	—	—	—
SL	50%	—	—	—	—	—	—	—
BJS	50%	—	—	—	—	—	—	—

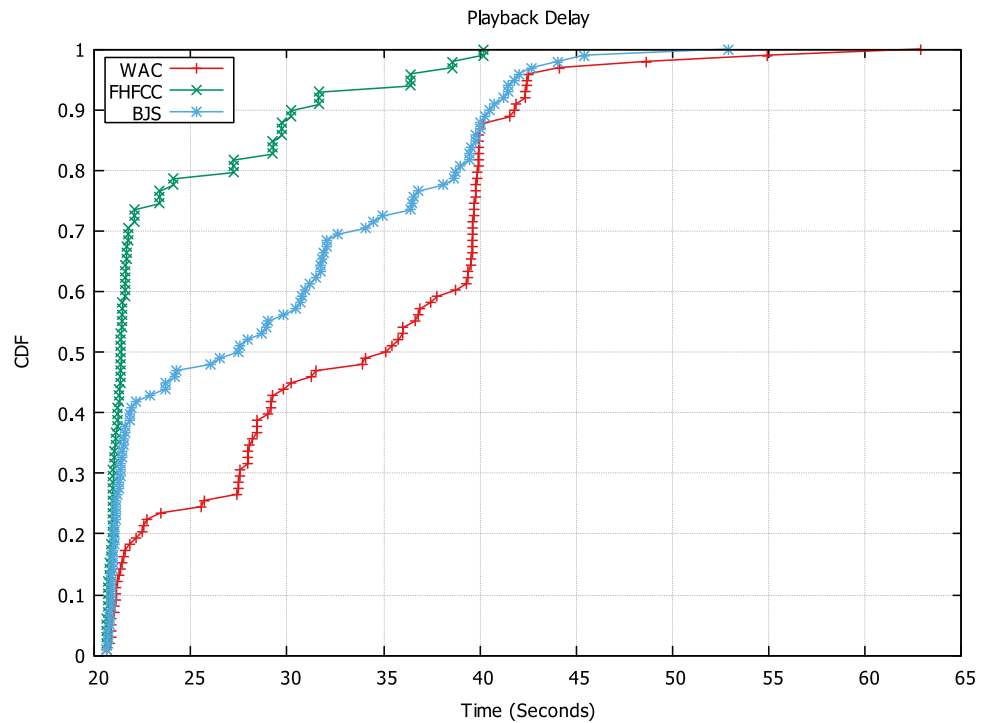
of more than 30 seconds. Figure 9 presents the Cumulative Distributed Function for start-up delay. On analyzing Fig. 9 it can be easily observed that 80% of peers in *FHFC*² are having less than 60 seconds start-up delay while in the case of without admission control approximately 35% of peers having more than 60 seconds of delay. In WAC, approx. 20% of peers are having more than 100 seconds delay, while some of the peers are having very high start-up delay approx. 200 seconds which degrades the overall quality and experience of the peer in the network. In BJS due to the batch-based entry of the peers' startup delay performs better as compared to the WAC approach. But due to improving congestion control *FHFC*² outperform the BJS. In BJS 20% of peers have more than 100 seconds startup delays. CDF for the total distortion is represented in Fig. 11. From the Figure, it is observed that in the case

of without admission control approximately 38% of peers have 100% distortion due to the flash crowd congestion. While in the case of *FHFC*² approx. 3% peers have 100% distortion. These are basically blocked peer due to the peers' selfish nature.

6.2 Discussion

Network without having any congestion control incurs significantly lower start-up delay, end-to-end delay, playback delay and total distortion as compared to all the other variants of congestion control due to the lower complexity. But as the number of peers increases, the network start to suffer from overload. Also, the network without admission control (WAC) suffer a huge amount of decrement in performance metrics which is shown in Table 5. In case of the *FHFC*²

Fig. 8 Playback delay with and without $FHFC^2$



approach, due to the fuzzy logic-based flash crowd prediction mechanism, the network is able to handle the overload and a significant improvement is observed in various parameters like start-up delay, playback delay, end-to-end delay and total distortion. The reason for the better performance of the proposed approach is due to the consideration of minimized requirement entry in primary mode, upload

bandwidth and utilization in overlay controller mode, efficient start-up buffer selection and quality-aware scheduler mode and score-based recovery mode. These factors optimize the selection of parent peers, efficient scheduling and fast recovery, which improves performance of the P2P live streaming under flash crowd. $FHFC^2$ approach not only improves the efficiency of the good peer but also motivates

Fig. 9 Start-up delay with and without $FHFC^2$

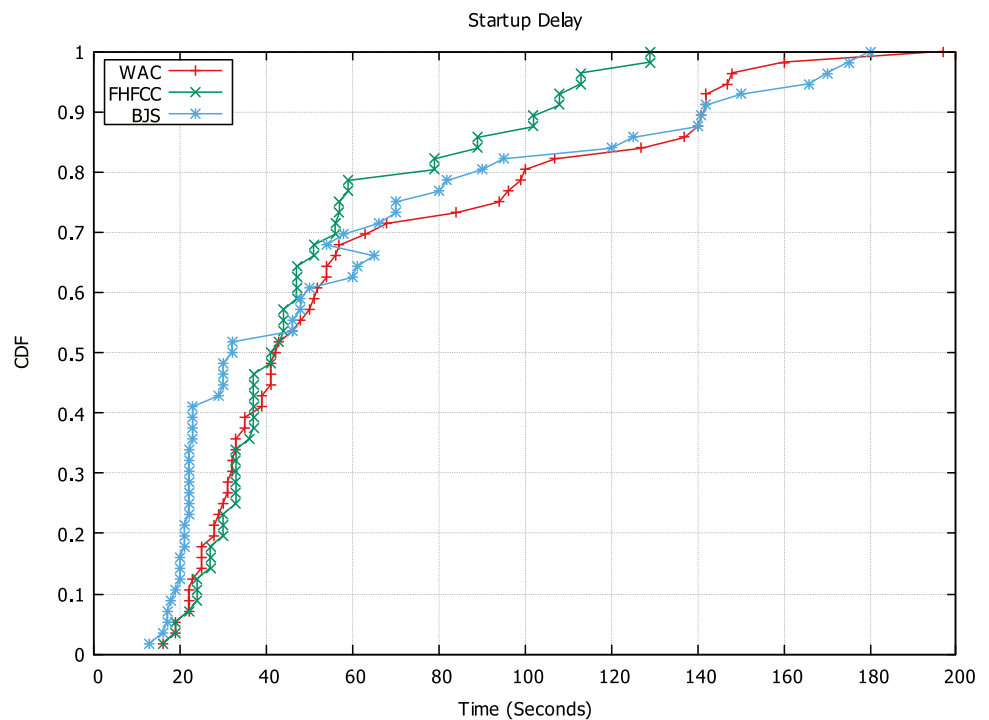
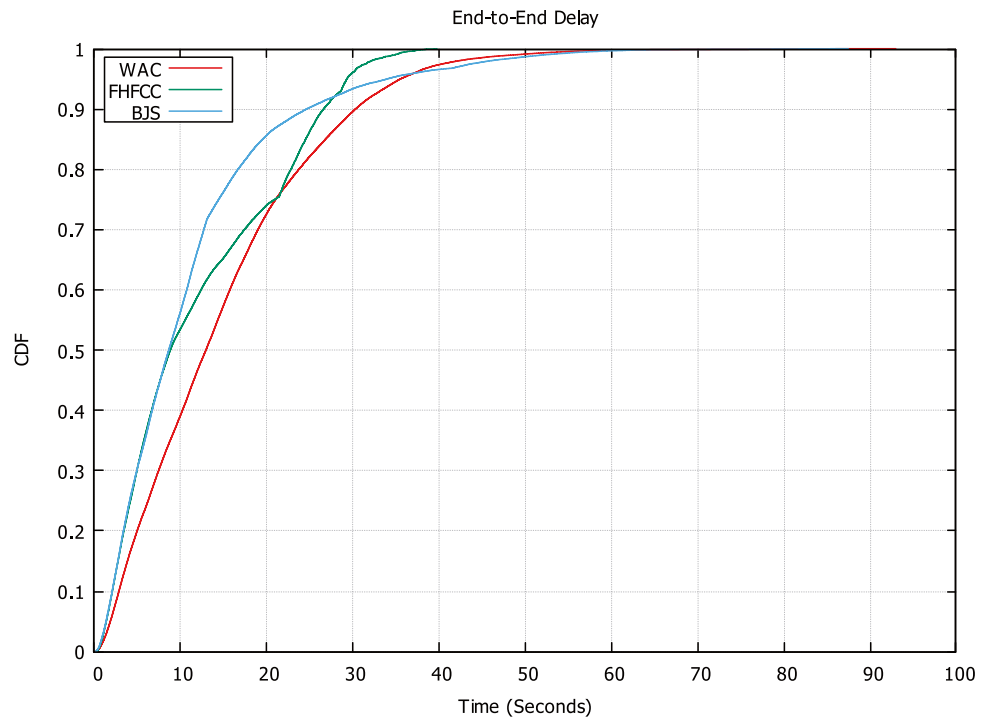


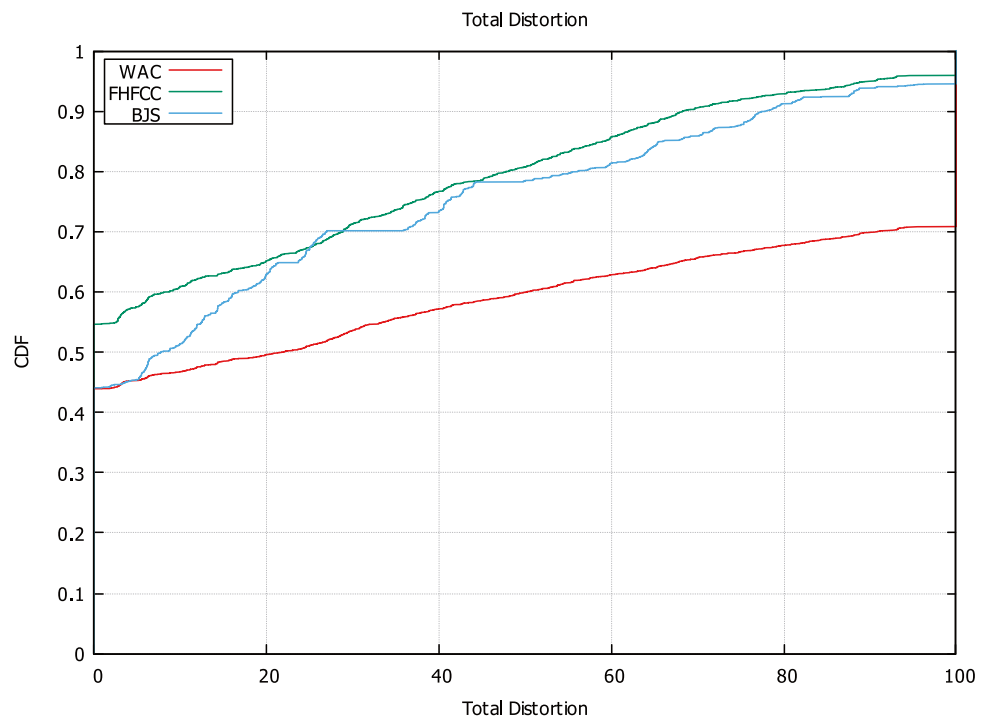
Fig. 10 End-to-End delay with and without $FHFC^2$



the selfish peer to participate actively in the network which is shown in Table 5. The $FHFC^2$ approach is scheduled and creates an overlay efficiently which helps to maintain the demand, and supply during the overload in the network and fulfill the basic requirement of available peers. Cumulative Distributed Function (CDF) curve is presented in Fig. 8 for a better understanding of the distribution of peers and their

respective parameters. From Fig. 8, it can be observed that as the number of peers increases, the playback delay of the peer increases, while in $FHFC^2$ making sure of the smooth handling of the peer in the network, which further helps to maintaining significant playback delay. Approx. 5% of peers in the network face significant playback delay due to the huge difference between demand and supply problem still,

Fig. 11 Total distortion with and without $FHFC^2$



*FHFC*² have a marginal increment in the playback delay as compared to those without admission control which is shown in Fig. 8. The reason behind the improved performance of selfish peers is that because of motivation packets lead to overall performance improvement of the P2P network.

So, as the level of flash crowd increases in the P2P live streaming hierarchical-based admission control handles it quite efficiently. If we compare it with centralized streaming system, still there are some improvement required. Centralized streaming services will perform better as compared to peer-to-peer live streaming systems. In flash crowd scenario, the performance of a centralized streaming systems will outperform peer-to-peer streaming systems due to their huge resource availability. Video quality and network performance of centralized systems are more, but the resource utilization and motivation of peers of P2P streaming is better as compared to centralized systems.

7 Conclusion and future work

In this paper, flash crowd and its dynamics in P2P-based live streaming systems have been studied. A fuzzy-based flash crowd prediction mechanism is developed to investigate the criticality of flash crowd by using the important factor - remain bandwidth, buffer size and event time. This has further led to propose a framework 'fuzzy-based hierarchical flash crowd controller' to control the congestion in the P2P networks. In this approach, the congestion controller is triggered to select the mode based on the predicted congestion value and subsequently, the congestion control mechanism scheme is activated. Several sub-algorithms are designed in each proposed mode. Contribution and reputation were also evaluated to identify selfish peers. Restriction, as well as motivation schemes, are also applied to victim peers. An overall improvement is observed when comparing the proposed algorithms with other variants. The overall complexity of the proposed approach is high. Our future work focuses on improving overall complexity and implementing the present work in real-time environment.

Author contributions All the authors have an equal contribution to making the manuscript an important research paper.

Funding For the research, authoring, and publication of this article, we have not received any financial funding.

Availability of data and materials The author(s) will provide the datasets and materials used and/or analysed in this work.

Code availability The author(s) will provide the code used in this work.

Declarations

Ethics approval This content is the authors' original work, which has never been published before.

Consent to participate It is affirmed that all the authors have seen and agreed to the submitted version of the paper and their inclusion of name(s) as co-author(s).

Consent for publication The author(s) hereby grants permission for the Work to be published.

Conflict of interest/Competing interests There are no conflicts of interest to disclose.

References

1. Venkataraman V, Francis P, Calandrino J (2006) Chunkyspread: Multi-tree unstructured peer-to-peer. In Proc. Int. Workshop on Peer-to-Peer Systems (IPTPS)
2. Jannotti J, Gifford DK, Johnson KL, Frans Kaashoek M, O'Toole Jr JW (2000). Overcast: Reliable multicasting with an overlay network. In Fourth Symposium on Operating Systems Design and Implementation (OSDI 2000)
3. Suphakit A, Su Z, Katto J (2010) ToMo: A two-layer mesh/tree structure for live streaming in P2P overlay network. In 2010 7th IEEE Consumer Communications and Networking Conference, pp 1-5. IEEE
4. Index, Cisco Global Cloud (2018) Forecast and methodology, 2016–2021 white paper. Updated
5. YouTube (2005) <https://www.youtube.com>. [Online; Accessed 20 Jan 2022]
6. IPTV (2007) howpublished = <https://www.iptvonline.ca/>. [online; Accessed 15 Jan 2022]
7. NETTV (2016) <http://nettv.com.np/nettv/>. [Online; Accessed 12 Jan 2022]
8. Alexa (1996) <http://www.alexa.com/topsites>. [Online; Accessed 15 Feb 2022]
9. Li B, Keung GY, Xie S, Liu F, Sun Y, Yin H (2008) An empirical study of flash crowd dynamics in a p2p-based live video streaming system. In IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference (pp. 1-5). IEEE
10. de Paula Junior U, Drummond LMA, de Oliveira D, Frota Y, Barbosa VC (2015) Handling flash-crowd events to improve the performance of web applications. In Proceedings of the 30th Annual ACM Symposium on Applied Computing, pp 769-774
11. Bhatia M, Rai MK (2017) Identifying P2P traffic: A survey. Peer-to-Peer Networking and Applications 10(5):1182–1203
12. Carbutaru C, Meng Teo Y, Leong B, Ho T (2013) Modeling flash crowd performance in peer-to-peer file distribution. IEEE Trans Parallel Distrib Syst 25(10):2617-2626
13. Chen Z, Li B, Keung G, Yin H, Lin C, Wang Y (2009) How scalable could P2P live media streaming system be with the stringent time constraint? In 2009 IEEE International Conference on Communications, pp 1-5. IEEE
14. Liu F, Li B, Zhong L, Li B, Jin H, Liao X (2011) Flash crowd in P2P live streaming systems: Fundamental characteristics and design implications. IEEE Trans Parallel Distrib Syst 23(7):1227–1239
15. D'Acunto L, Vinkó T, Sips H (2011) Bandwidth allocation in bittorrent-like vod systems under flashcrowds. In 2011 IEEE International Conference on Peer-to-Peer Computing, pp. 192-201. IEEE
16. Bayat N, Lutfiyya H (2015) Network coding for coping with flash crowd in P2P multi-channel live video streaming. In 2015 11th International Conference on the Design of Reliable Communication Networks (DRCN), pp 243-246. IEEE
17. Xie L, Smith P, Hutchison D, Banfield M, Leopold H, Jabbar A, Sterbenz JPG (2008) From detection to remediation: a self-organized system for addressing flash crowd problems. In 2008

- IEEE International Conference on Communications, pp 5809–5814. IEEE
18. Xu G, Hu Y, Shen Y, Guo M (2012) Directedpush-a high performance peer-to-peer live streaming system using network coding. In 2012 IEEE 18th International Conference on Parallel and Distributed Systems, pp 292–298. IEEE
19. Chen Y, Zhang B, Chen C (2011) Modeling and performance analysis of P2P live streaming systems under flash crowds. In 2011 IEEE International Conference on Communications (ICC), pp 1–5. IEEE
20. Liu F, Li B, Zhong L, Li B, Niu D (2009) How P2P streaming systems scale over time under a flash crowd? In IPTPS 1:2
21. Wu H, Xu K, Zhou M, Wong AK, Li J, Li Z (2013) Multiple-tree topology construction scheme for P2P live streaming systems under flash crowds. In 2013 IEEE wireless communications and networking conference (WCNC), pp 4576–4581. IEEE
22. Chung TY, Lin O (2011) A batch join scheme for flash crowd reduction in iptv systems. In 2011 IEEE 17th International Conference on Parallel and Distributed Systems, pp 823–828. IEEE
23. Chen Y, Zhang B, Chen C, Chiu DM (2013) Performance modeling and evaluation of peer-to-peer live streaming systems under flash crowds. *IEEE/ACM Trans Netw* 22(4):1106–1120
24. Hefeeda M, Saleh O (2008) Traffic modeling and proportional partial caching for peer-to-peer systems. *IEEE/ACM Trans Netw* 16(6):1447–1460
25. BitTorrent (2019) <https://www.bittorrent.com/>. [Online; Accessed on 05- Feb 2022]
26. Ripeanu M (2001) Peer-to-peer architecture case study: Gnutella network. In Proceedings First International Conference on Peer-to-Peer Computing, pp 99–100. IEEE
27. Karagiannis T, Broido A, Brownlee N, Claffy K, Faloutsos M (2003) File-sharing in the Internet: A characterization of P2P traffic in the backbone. University of California, Riverside, USA, Tech Rep
28. Heckmann O, Bock A (2002) The edonkey 2000 protocol. Rapport technique, Multimedia Communications Lab, Darmstadt University of Technology 13
29. Lingjun M, Tsang P-S, Lui K-S (2009) Improving file distribution performance by grouping in peer-to-peer networks. *IEEE Trans Netw Serv Manag* 6(3):149–162
30. Shen H (2009) An efficient and adaptive decentralized file replication algorithm in P2P file sharing systems. *IEEE Trans Parallel Distrib Syst* 21(6):827–840
31. Li M, Jiadi Y, Jie W (2008) Free-riding on bittorrent-like peer-to-peer file sharing systems: Modeling analysis and improvement. *IEEE Trans Parallel Distrib Syst* 19(7):954–966
32. Lee U, Park J-S, Lee S-H, Ro WW, Pau G, Gerla M (2008) Efficient peer-to-peer file sharing using network coding in MANET. *J Commun Networks* 10(4):422–429
33. Farrelly B, Sun Y, Mahanti A, Gong M (2017) Video workload characteristics of online porn: Perspectives from a major video streaming service. In 2017 IEEE 42nd Conference on Local Computer Networks (LCN), pp 518–519. IEEE
34. Song Y-D, Gong M, Mahanti A (2019) Measurement and analysis of an adult video streaming service. In 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp 489–492. IEEE
35. Yu R, Christophersen C, Song Y-D, Mahanti A (2019) Comparative analysis of adult video streaming services: characteristics and workload. In 2019 Network Traffic Measurement and Analysis Conference (TMA), pp 49–56. IEEE
36. Song Y-D, Mahanti A (2019) Comparison of mobile and fixed device workloads in an academic web server. In 2019 IEEE International Symposium on Measurements & Networking (M & N), pp 1–6. IEEE
37. Mahanti A, Carlsson N, Arlitt M, Williamson C (2012) Characterizing cyberlocker traffic flows. In 37th Annual IEEE Conference on Local Computer Networks, pp 410–418. IEEE
38. Mahanti A, Carlsson N, Mahanti A, Arlitt M, Williamson C (2013) A tale of the tails: Power-laws in internet measurements. *IEEE Netw* 27(1):59–64
39. Ansari SA, Pal K, Govil MC, Ahmed M, Chawla T, Choudhary A (2021) Score-based Incentive Mechanism (SIM) for live multimedia streaming in peer-to-peer network. *Multimed Tools Appl* 80(13):19263–19290
40. Ansari Md, Alam S, Pal K, Govil MC, Govil P, Srivastava A (2021) Ensemble Machine Learning for P2P Traffic Identification. *International Journal of Computing and Digital Systems*
41. Miguel EC, Cunha Í, Silva CM, Carvalho F, Campos SVA (2017) Resource-constrained P2P streaming overlay construction for efficient joining under flash crowds. In 2017 IEEE Symposium on Computers and Communications (ISCC), pp 639–644. IEEE
42. Zheng Q, Long Y, Qin T, Yang L (2011) Lifetime characteristics measurement of a P2P streaming system: Focusing on snapshots of the overlay. In 2011 9th World Congress on Intelligent Control and Automation, pp 805–810. IEEE
43. Meulpolder M, Meester LE, Epema DHJ (2013) The problem of upload competition in peer-to-peer systems with incentive mechanisms. *Concurrency and Computation: Practice and Experience* 25(7):899–917
44. Payberah AH, Dowling J, Haridi S (2011) Glive: The gradient overlay as a market maker for mesh-based p2p live streaming. In 2011 10th International Symposium on Parallel and Distributed Computing, pp 153–162. IEEE
45. Lobb RJ, da Silva APC, Leonardi E, Mellia M, Meo M (2009) Adaptive overlay topology for mesh-based P2P-TV systems. In Proceedings of the 18th international workshop on Network and operating systems support for digital audio and video, pp 31–36
46. Wu H, Liu J, Jiang H, Sun Y, Li J, Li Z (2012) Bandwidth-aware peer selection for P2P live streaming systems under flash crowds. In 2012 IEEE 31st International Performance Computing and Communications Conference (IPCCC), pp 360–367. IEEE
47. da Silva APC, Leonardi E, Mellia M, Meo M (2008) A bandwidth-aware scheduling strategy for P2P-TV systems. In 2008 Eighth International Conference on Peer-to-Peer Computing, pp 279–288. IEEE
48. Basher N, Mahanti A, Mahanti A, Williamson C, Arlitt M (2008) A comparative analysis of web and peer-to-peer traffic. In Proceedings of the 17th international conference on World Wide Web, pp 287–296
49. Wu H, Jiang H, Liu J, Sun Y, Li J, Li Z (2011) How P2P live streaming systems scale quickly under a flash crowd?. In 30th IEEE international performance computing and communications conference, pp 1–8. IEEE
50. Wallach DS (2002) A survey of peer-to-peer security issues. In International symposium on software security, pp. 42–57. Springer, Berlin, Heidelberg
51. Androutsellis-Theotokis S, Spinellis D (2004) A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys (CSUR)* 36(4):335–371
52. Pal K, Govil MC, Ahmed M (2018) Slack time-based scheduling scheme for live video streaming in P2P network. *Int J Commun Syst* 31(2):e3440
53. Shen H, Lin Y, Li J (2014) A social-network-aided efficient peer-to-peer live streaming system. *IEEE/ACM Trans Netw* 23(3):987–1000
54. Huang S, Izquierdo E, Hao P (2016) Bandwidth-efficient packet scheduling for live streaming with network coding. *IEEE Trans Multimedia* 18(4):752–763
55. Pal K, Govil MC, Ahmed M (2018) Priority-based scheduling scheme for live video streaming in peer-to-peer network. *Multimed Tools Appl* 77(18):24427–24457
56. Maheswari BU, Ramesh TK (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
57. Pal K, Govil MC, 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) 41:431–437. IEEE
58. Pal K, Govil MC, Ahmed M (2019) FLHyO: fuzzy logic based hybrid overlay for P2P live video streaming. *Multimed Tools Appl* 78(23):33679–33702
 59. Gupta Ankur, Awasthi Lalit K (2011) Peer-to-peer networks and computation: Current trends and future perspectives. *Computing and Informatics* 30(3):559–594
 60. 47 Samuvelraj G, Nalini N (2014) A survey of self organizing trust method to avoid malicious peers from peer to peer network. In 2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE), pp 1–4. IEEE
 61. Li B, Yin H (2007) Peer-to-peer live video streaming on the internet: issues, existing approaches, and challenges [peer-to-peer multimedia streaming]. *IEEE Commun Mag* 45(6):94–99
 62. Shang-Fu G, Jian-Lei Z (2012) A survey of reputation and trust mechanism in peer-to-peer network. In 2012 International Conference on Industrial Control and Electronics Engineering, pp 116–119. IEEE
 63. Ansari Md, Alam S, Pal K, Govil P, Govil MC, Awasthi LK (2022) A statistical analysis of SAMPARK dataset for peer-to-peer traffic and selfish-peer identification. *Multimed Tools Appl*, pp 1–29
 64. Lua EK, Crowcroft J, Pias M, Sharma R, Lim S (2005) A survey and comparison of peer-to-peer overlay network schemes. *IEEE Commun Surv Tutor* 7(2):72–93
 65. 39 Park H, Yang J, Park J, Kang SG, Choi JK (2008) A survey on peer-to-peer overlay network schemes. In 2008 10th international conference on advanced communication technology 2:986–988. IEEE
 66. Zhang J, Zhang Y, Shen M (2019) A distance-driven alliance for a p2p live video system. *IEEE Trans Multimedia* 22(9):2409–2419
 67. Type AM, Logic F (1979) A Mamdani Type Fuzzy Logic Controller
 68. Marchant T (2007) A measurement-theoretic axiomatization of trapezoidal membership functions. *IEEE Trans Fuzzy Syst* 15(2):238–242
 69. Iancu I (2012) A Mamdani type fuzzy logic controller. *Fuzzy logic-controls, concepts, theories and applications* 15(2):325–350
 70. Baumgart I, Heep B, Krause S (2007) OverSim: A flexible overlay network simulation framework. In 2007 IEEE global internet symposium, pp 79–84. IEEE

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.



Md. Sarfaraj Alam Ansari has received M. Tech degree in Information Security from National Institute of Technology Durgapur, India and PhD in Computer Science & Engineering from National Institute of Technology Sikkim, India. He is working as Assistant Professor at National Institute of Technology Sikkim, India. He is having more than 03 years of industry experience and 10 years' experience in academics and research. His research interest includes Computer Networks and communications, Information

Security & Risk Management, Cloud Computing, etc.



Kunwar Pal has received B. Tech in CSE from KNIT, Sultanpur, India in 2009, M. Tech (CSE) from PEC, Chandigarh, India in 2011 and his Doctorate from MNIT Jaipur, in 2018. Presently he is working as an Assistant Professor at Dr. B R Ambedkar NIT, Jalandhar. His research interests include Peer-to-peer network, Multimedia Communication, Network Security, real-time system etc.



Prajval Govil is working as an Associate Data Scientist at Celebal Technologies, Jaipur, Rajasthan, India. He has received his B.Tech from JK Lakshmipat University, Jaipur, in 2021. His research interest includes Machine Learning, Deep Learning, Big data Analytics, Data Science. He has completed more than 10 undergraduate projects on the topic mentioned above. He is also engaged in various client related projects on the above-mentioned topics and has currently 8 months industry experience.



Mahesh Chandra Govil is a Professor in the Department of Computer Science and Engineering at Malviya National Institute of Technology Jaipur – Rajasthan, India. Currently, he is the Director at the National Institute of Technology Sikkim, Sikkim, India. He received his Ph.D. in Computer Science and Engineering from the Indian Institute of Technology Roorkee, Uttarakhand, India, in 2003. His current

research interests include Real-Time Systems, Image Processing, Distributed Systems, Fault Tolerance in Cloud Computing, etc.