Journal of Network and Systems Management (2019) 27:815–836
https://doi.org/10.1007/s10922-018-09485-6

Collaborative Caching in P2P Streaming Networks

Guoqiang Gao[1] · **Ruixuan Li**[2]

Received: 8 October 2016 / Revised: 25 November 2018 / Accepted: 8 December 2018 /
Published online: 4 January 2019

Abstract

With the development of the Internet, the Internet applications represented by Peer-to-Peer (P2P) streaming systems bring more and more communication overhead. In order to alleviate communication pressure of the Internet, and improve the user's access experience, we aim at taking full advantage of peer upload bandwidth contributions with a cache on each peer. While, the size of such a cache on each peer is limited, it is necessary to design the cache replacement strategy in such P2P Systems. In this paper, we implemented a collaborative caching mechanism for P2P streaming networks. Firstly, we designed some measurement experiments for the existing large-scale P2P streaming systems to find some features and problems. Then, we proposed the collaborative caching strategy based on the chunk value and requesting distribution factor to address the existing problems. Our goal is to keep those chunks with high requesting frequency and farther away from the current peer in order to achieve higher cache hit rate and load balancing. Compare to the measured P2P streaming systems, the simulation results show that the proposed method has better performance on multiple metrics such as cache hits, system load and peer collaboration.

Keywords P2P measurement · Traffic · Distribution · Replacement · Chunk value

✉ Guoqiang Gao
ggao@wtu.edu.cn

Ruixuan Li
rxli@hust.edu.cn

[1] School of Media and Communication, Engineering Research Center of Hubei Province for Clothing Information, Wuhan Textile University, No.1, Fangzhi Road, Hongshan District, Wuhan 430073, Hubei, People's Republic of China

[2] Intelligent and Distributed Computing Laboratory, School of Computer Science and Technology, Huazhong University of Science and Technology, No.1037, Luoyu Road, Wuhan 430074, Hubei, People's Republic of China

1 Introduction

The Internet has been tremendous growth in the last 10 years. According to the report of eMarketer [1], more than half of the world's population will use the internet regularly in 2019. With the increase of Internet users, distributed applications, such as P2P file sharing, distributed computing, Internet telephony, Internet streaming media and online social networks, etc., have been gaining popularity. Among all these applications, Internet streaming and P2P file sharing systems are the killing applications which occupy the largest percentage of the world wide Internet traffic. In the reference [2], Cisco forecasts that global Internet Protocol (IP) traffic will reach 1.1 ZB (zettabyte: 1000 exabytes) per year in 2016 and will exceed 2.3 ZB by 2021. The report of Cisco shows that IP video traffic now accounts for 73% of all consumer Internet traffic in 2016, and P2P related traffic accounts 30% of it. The thing is even worse that these two applications are still the fastest growing applications. By 2021, the video related applications (P2P and Internet streaming) will account for more than 82% of the total IP traffic. Clearly, those applications generate higher and higher pressures on Internet backbone and service providers.

For distributed applications, the cache can effectively reduce the transmission of data on the Internet backbone and improve the user access experience. In the network applications, the cache is a key factor to achieve effective network services, and it has been widely used by many organizations, such as universities, government agencies, companies and Internet service providers (ISP). The most popular caching mechanism [3–5] is web caching which uses the dedicated servers to save the cache files. In those applications, the dedicated proxy servers are deployed in the edge of the network close to the user. If an object is requested by a client, the proxy server will generate a copy for it. Therefore, for the same following requests, this object can be directly obtained from the proxy server, which can reduce transmission overhead. To relieve Internet communication pressure given by P2P applications, industry and academia have proposed many solutions to apply web caching to P2P systems [6, 7]. Like web caching, most of those strategies are deploying dedicated caching servers at ISP edge or network boundaries, which can ensure that all incoming and outgoing P2P objects are cached. When a client queries a cached object, it can directly obtain the object from the cache servers rather than from the original peer. Although this mechanism can make use of the advantages of web caching, there are some problems. Firstly, it will lead to high investment costs because each ISP has to purchase and install expensive proxy servers. Secondly, applying web caching to P2P will make the proxy servers be hot spots and bring single point of failure problem. This is because all P2P data must pass through them. Once the servers fail, the cache mechanism will fail. Finally, the caching space of the proxy server is limited. P2P streaming like applications usually share audio and video files whose sizes range from a few megabytes to hundreds of megabytes, even up to several GB (gigabyte), and those objects' size is much larger than the files in web caching.

By using the storage space of normal peers in P2P networks to cache data, we can greatly improve the efficiency of the system and relieve traffic pressure on the

Internet. This mechanism is called P2P distributed caching. In order to design efficient caching scheme in P2P streaming systems, we measure some existing P2P streaming systems firstly, such as PPTV [8] and iQIY [9]. The experimental results show that the two measured P2P streaming systems in China have better viewing experience, especially for Chinese users. After further analysis of the collected data, we found that both PPTV and iQIY all use the hybrid caching mechanism, which is the combination of web caching and P2P distributed caching. In order to provide better service, they deploy a large number of dedicated servers in China to cache channel data they provide. When a peer cannot get data from other peers, it can download from the dedicated servers. Thus the system can provide a desired downloading rate. This strategy is similar to web caching, and has the problems we discussed above. Furthermore, for those areas without the dedicated servers, the user will not be able to benefit from this strategy, such as outside China. We also found that the cache utilization at each peer is very low. Therefore, how to take full advantage of the efficiency of P2P distributed caching is worthy of study. To solve the problems we discussed above, we propose a collaborative caching mechanism for P2P streaming systems. In data replacement, we not only consider the request frequency of the data, but also consider the distribution of the data location, that is a peer caches the data farther away from it as possible. The goal of our strategy is that the cached data is evenly distributed to the network peers, and providing better data download rate in the case of lack of the dedicated servers.

The remaining sections of this paper are organized as follows. Section 2 describes related work on P2P streaming. In Sect. 3, we discuss the measurement experiments on some popular P2P streaming systems, and analyze the results. Section 4 introduces the detail of our collaborative caching mechanism, and in Sect. 5 we present the simulation experiments and results analysis for proposed strategy. Finally, we summarize this paper, and propose further research work.

2 Related Work

Most of commercial P2P streaming applications, such as PPTV, iQIY, QQLive [10], UUSee [11], and SopCast [12], all use the mesh-based P2P architecture. In those systems, the viewers of a channel come from the same overlay network. The video data of each channel is divided into many segments, and then further divided into a number of chunks [13]. The video source server is responsible for the initialization of a video broadcast, which is the source of a video program. If a user want to watch a video channel in P2P streaming system, his client will join the P2P network of the channel. The client not only receives the video data from both the video source servers and other clients to play programs, but also becomes a relay peer for video dissemination. An effective P2P streaming system should ensure that the video data is exchanged between the peers periodically, and only downloaded from the video source servers in rare cases.

To relieve the burden imposed by P2P traffic, design and implement an effective caching infrastructure in P2P streaming systems attracted great interests from

both industry and academia [14–16]. However, it is difficult due to the unique features such as self-governing, and dynamic membership, large number of peers, and even larger amount of shared files in P2P applications. Wierzbicki et al. discussed different features of P2P traffic and web traffic in [17]. They proposed specialized cache replacement policies such as MINRS (their cache replacement policy called Minimum Relative Size) and LRSB (the other one called Least Relative Sent Bytes). They also conducted simulation experiments to evaluate the performance of various replacement policies for fasttrack traffic. However, they still used the same architecture as web caching by only considering caching policies for a dedicated proxy server (providing cache services for peers within the boundary of an Intranet). The clients themselves are not participated in the caching service. In [18], Hefeeda et al. proposed similar idea by deploying caches at or near the border of the ASs (autonomous systems), that is called pCache. pCache will intercept P2P traffics go through the AS, and try to cache the most popular contents. The cache size is relatively small, and the objects in P2P applications are very big, so the effectiveness of this approach is doubtable. Furthermore, pCache itself became a bottleneck and a single point of failure, which will affect its efficiency.

SECC [19] introduces a novel index structure called Signature Augment Tree (SAT) to implement collaborative caching. Simulation results show that SECC schemes significantly outperform other caching methods which are based on existing spatial caching schemes in a number of metrics, including traffic volume, query latency and power consumption. However SECC is only suitable for mobile P2P networks. In PECAN [20], each peer adjusts its cache capacity adaptively to meet the server's upload bandwidth constraint and achieve the fairness. Dai et al. [21] have developed a theoretical framework to analyze representative cache resource allocation schemes within the design space of collaborative caching, with a particular focus on minimizing costly inter-ISP traffic. Their researches not only help themselves and other researchers understand the traffic characteristics of existing P2P streaming systems in light of realistic elements, but also offer fundamental insights into designing collaborative ISP caching mechanisms. We use research ideas in our study of P2P streaming distributed caching. Firstly, we study existing P2P streaming systems through measurement experiments to find some problems. Then we propose some solutions for the problems. In this paper, we found some problems in the existing P2P streaming systems by using measurement research method. Meanwhile, we also found some useful network characteristics. To study the existing P2P streaming system, P2P network measurement and analysis, valuable information, provide support for further improvement of the network and design new algorithms, which is very meaningful issue. These findings are useful for further improvement of the P2P streaming systems, and can even help us to generate new ideas in the algorithms design.

To better understand user behaviors, network characteristics and performance bottlenecks in P2P streaming applications, numerous research papers have been published to evaluate and analyze these bandwidth intensive applications [22–24]. Jeff [25] conducted an experimental comparison of two streaming applications (a mesh-based and a multiple tree-based). This was the first study to compare streaming overlay architectures in real Internet settings, considering not only intuitive

aspects such as scalability and performance under churn, but also less studied factors such as bandwidth and latency heterogeneity of overlay participants. In their study, the authors found that mesh-based approach is superior to tree-based approach. Their conclusion could explain why most commercial P2P streaming applications are using mesh-based infrastructure. However, they conducted the controlled small-scale measurement on PlanetLab [26], which only has a small number of concurrent users, and they chose Chainsaw [27] and SplitStream [28] systems which were not the real world commercial systems. Mu [29] deploys a P2P-based IPTV (Internet Protocol Television) system for thousands of students, and then presents a multi-modal QoE measurement framework which evaluates the IPTV services by collaborating measurements with a variety of different aspects. Maybe they can get better results. Bing [30] developed a piece of measurement software, named as VoD-Crawler, to implement their experiments for PPLive. They have revealed the major features of the P2P VoD (video on demand) overlay networks and have compared them with those in P2P file sharing and live streaming systems. However, they did not further discuss how to improve P2P streaming systems. Ciullo et al. [31] conducted a controlled measurement evaluation on PPLive and Joost [32]. The authors evaluated the characteristics of both data distribution and signaling process for the overlay network discovery and maintenance. By considering two or more clients in the same subnetwork, they observed the capability of both systems to exploit the locality of peers. They also explored how the system adapts to different network conditions. In this paper, we estimate the characteristics of P2P streaming system by measuring the network traffic of its peers. This method is simple and feasible. We will introduce it in detail in the next section.

3 Measurement on P2P Streaming Systems

Almost all of the large-scale commercial P2P streaming systems are not open source software. In order to analyze and summarize their features, we use the measurement method to study them. In this section, we measure two popular P2P streaming systems in China, PPTV and iQIY, to discover some of the characteristics and problems from them. Based on our findings, we can design P2P streaming collaborative caching better.

3.1 Measurement Methodology

For traffic, we use WireShark [33] for trace data collection. WireShark is similar to tcpdump [34], except that it has a graphic front-end interface. It is easy to use WireShark and set it up with different configurations. We can create a filter to ignore uninterested data packages, decide the time duration to capture packages, and choose single file or multiple files to store the data. To collect trace, we run both WireShark and P2P streaming clients simultaneously. WireShark records all packages passing through the experimental machines into binary files. Our measurement objects are two of the most popular P2P streaming systems, namely, PPTV and iQIY. Since the

data and signal transmission are mainly done with Transfer Control Protocol (TCP) and User Datagram Protocol (UDP) protocols, we only keep these two types of packets in our experiments. The structure of IP frame is public so that we can obtain valuable information from this part. For example, we can extract the IP address of the source peers by reading the package frame header. The data load carried by IP frame can be further divided into P2P streaming system header and video data. As PPTV and iQIY are not the open-source applications, we do not have their header structure information available. However, we can still observe some useful information by analyzing a large amount of trace data. For example, we found that the length of PPTV header is 52bytes, and the bytes 7–9 indicate the sequence number. The sequence number identifies a data block location in the entire video and can be used for cache performance analysis.

In order to analyze the caching mechanism of P2P streaming systems, we develop a real-time monitoring program by using the Perl language. Our program can periodically scan the cache space of P2P streaming systems to observe the changes of the cached data, which can help us to estimate the caching strategy of measured P2P systems. Perl language is a high-performance scripting language, and it is simple to implement. Our Perl code automatically detects the cache space of P2P applications every 5 s, and records the changes of cached objects. For PPTV, it uses a cache directory to cache downloaded video clips. For example, "Avatar01.mp4" indicates it is the first video clip of the channel "Avatar" (Avatar is a VoD channel. In most cases, a VoD channel is to provide a video to users in the P2P streaming system.), and its size is usually 20M. Our code will record the video files in PPTV cache directory in each scanning, and the record format is <time, filename>. For iQIY, it caches its downloading data into a big file whose structure is more complex than PPTV. We understand its structure after a lot of analysis and write a shell to obtain caching information from it just like PPTV.

To analyze the collected data, we use a two-tuple $\{IP_A, IP_B\}$ to represent a flow between our clients A and another host B, where IP_A and IP_B are the IP address of host A and B respectively. The flow has directionality such as the two flows $\{IP_A, IP_B\}$ and $\{IP_B, IP_A\}$ are treated as two separate flows. Assuming $F(A, B) = \{IP_A, IP_B\}$ is a flow between hosts A and B, the size of a packet that is part of this flow can be represented as $P_s(A, B, t)$ where t is the timestamp at which the packet is recorded. Therefore, the rate of host A for a channel can then be represented at Eq. 1 that will be used in the experiments.

$$R(A) = \frac{\sum_{\{X \text{ in } List_A\}} \sum_{t=t1}^{t1+\Delta} P_s(A, X, t)}{\Delta} \tag{1}$$

where $List_A$ is a set of the hosts communicating with host A. In our experiment evaluation, we use a Δ of 60 s, so that bandwidth is calculated on non-overlapping intervals of 60 s each. In order to obtain the request graph of cached data, we define $P_i(A, B)$ to represent the packet with sequence number i between host A and B for a channel. The packets' number of traffic provided by host A with sequence number i for a channel can then be represented as

$$P_i(A) = \sum_{\{X \ in \ List_A\}} P_i(A, X) \tag{2}$$

After obtaining all $P_i(A)$, $i = 1 \ldots n$, for a channel, we can draw the requested curve for this channel, which can be used to reveal the feature of measured applications in uploading.

As for the collected data related the caching replacement, we use a two-tuple $\{V_i, t\}$ to represent the caching trace data where V_i is the basic cached video unit such as "sanshengsanshi(01)[1].mp4" in PPTV and t is the recording time for V_i. Because the caching directory or file is recorded at every 10 min, we can consider the recorded video fragment is kept for 10 min in the caching buffer. We use $G(V_i, t) = 10$ to represent the cached time of V_i is 10 min. Therefore, the keep time of V_i in the caching buffer can then be represented as

$$T(V_i) = \sum_{t=t_{start}}^{t_{end}} G(Vi, t) \tag{3}$$

where t_{start} is the start time of caching trace collection and t_{end} is the end time. For a channel, we can draw the keep time curve of its cached video files after collecting all $T(V_i)$. Thus, we can estimate the cache replacement policy of the measured P2P streaming applications.

3.2 Measurement Channels

PPTV and iQIY are currently very popular P2P streaming applications in China, and they have a huge number of users. We can find some valuable information by measuring them. Our measuring clients are two high-performance Dell PowerEdge T630 servers. Their CUP is Intel Xeon E5-2620, and memory is 16G. The two computers run PPTV and iQIY respectively, and are connected to the network: China Telecom. The bandwidth of each computer is assigned to 10Mbps. The measurement channels are video-on-demand (VoD) channels. For each P2P streaming system, we choose two channels, one for the popular channel, the other is unpopular. The chosen channels are different for each category because the program libraries of PPTV and iQIY are different. We choose whether the channel is popular or not based on the rules of the two measured applications. The VoD channels we choose are all episode-based TV Series channels, and each of them consists of multiple episodes with the number varying from 44 to 86.

Table 1 shows the information about the VoD channels we use, where the admission is the number of times a channel is watched. For each channel, we collect both traffic data and cache data. In order to express more clearly in comparison, the measurement results are normalized to [0,100] in some measurement analysis because the lengths of the channels are different. The popular Chinese TV series "Eternal Love" and the unpopular Kongkong channel "Concubine of Qing Emperor" are chosen for PPTV. In order to clearly present the experimental result, each channel is used its first letters to represent it, such EL represents the channel "Eternal Love" and CQE

Table 1 List of measurement channels

Channel name	Popularity	Episodes	Admissions (M)
PPTV			
Eternal Love (EL)	Popular	58	2370
Concubine of Qing Emperor (CQE)	Unpopular	44	20
iQIY			
The Princess Weiyong (PW)	Popular	54	9190
Dae Wang Sejong (DWS)	Unpopular	86	3

is "Concubine of Qing Emperor". For iQIY, the popular Chinese channel "The Princess Weiyong" and the unpopular Korean channel "Dae Wang Sejong" are selected to measure. The detail information is shown in Table 1.

3.3 Experimental Results

We found some useful characteristics of the large-scale P2P streaming applications based on analysis and comparison on collected data. The evaluation metrics include the total amount of data received, cache replacement and peer geographic distribution. In this section, we will present detailed measurement information.

3.3.1 Traffic Analysis

First, we introduce the total amount of video data transmitted. Table 2 summarizes the results for both PPTV and iQIY channels. The second and fifth columns show the sum of video traffic received for each channel. The third and sixth columns indicate the percentage of data received from the dedicated servers. We found, for both PPTV and iQIY, the total size of received data for each channel is approximately equal to the product of its playback period and its bitrate. Therefore, all the channels have good viewing experience, whether they are popular or unpopular programs. For the popular channel, it seems reasonable based on P2P basic principle. However, how to achieve this effect for the unpopular channels? From Table 2, we can learn this reason that both PPTV and iQIY use the dedicate servers to improve the unpopular channels performance. In experiments, we identify a peer as a dedicated

Table 2 List of total amount of data downloaded

PPTV			iQIY		
Channel	Total received (MB)	Dedicated server (%)	Channel	Total received (MB)	Dedicated server (%)
EL	9019	0.52	PW	9805	1.02
CQE	5297	67.80	DWS	11,870	73.15

server if its IP frequently appear at most of channels in the entire measurement period. About 70% data is downloaded from the dedicated servers for the channel CQE and DWS. While, the popular channel EL only downloads 0.52% data from the dedicated servers. Although the deployment of dedicated servers can improve performance, it will degenerate into the client/server (C/S) model whose working principle is that the client only obtains data from the server. Thus, this strategy has some C/S inherent defects, such as server overhead, bottleneck, poor scalability, and etc. Furthermore the dedicated servers deployed in China cannot satisfy all users, such as the overseas users.

3.3.2 Peer Collaboration

The principle of P2P paradigm is to build a tight cooperation relationship among participating peers. We analyze P2P streaming systems' performance in terms of peer receive/send data ratio. We can use Eq. 1 to obtain this data. In order to show peer collaboration, we list the top 100, 500 and all peers that our clients have received video data from them, and they are called Top 100 peers, Top 500 peers and All peers respectively in the analysis of the experimental results. For the measured channels, the number of all peers, that is the peers that has been connected to our measurement clients, is 10 K to 3 M. The number of all peers of the popular channels is much more than the unpopular channels. For example, for the popular channel EL of PPTV, the number of all peers is 3 M, however the unpopular channel CQE is only 10 K. Based on the collected data, we count the ratio of the total data our client sends to these peers over all sent traffic. The results are shown in Fig. 1. There is very few peer cooperation for both PPTV and iQIY. For all channels, only 0.5% to 16% of sent data traffic goes to the peers that our client have received data

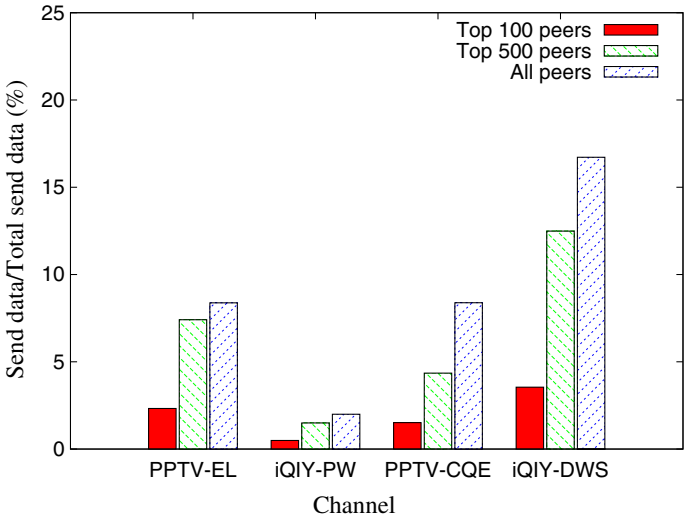

Fig. 1 Peer receive/send data ratio

from them. In these channels, when our clients watch an episode, it always retrieves the video content from those peers who have already watched this part and cached it on the disk. Most likely, these peers are way in advance. The video content needed by them cannot be retrieved from our client. They can only request from other peers who are watching ahead of them. Thus, very few data exchange occurs. This problem is even worse for the most unpopular channel CQE since its data is mainly retrieved from the dedicated servers. Servers do not retrieve data from peers. This scenario does not mean bad resource utilization among peers. It only represent that, in these channels, the group of peers our client receiving data from has little overlap with the group of peers our client sending data to. Some mechanisms can be used to improve the performance such as prefetching whose mechanism is to prepare the data for the coming reading. We plan to do further investigations on this issue.

3.3.3 Cache Replacement

In order to understand the caching strategy of P2P streaming systems, we design an experiment to measure the replacement strategy of the caching. In this experiment, we set PPTV cache size to 1024M, and collect the keep time of video files in PPTV cache directory. As for iQIY, its cache size is set to 1024M as well, and the keep time of video files is obtained from its cache files. Based on the collected data, we can estimate the caching replacement strategies of PPTV and iQIY, the results are shown in Fig. 2. To clearly describe the results, we only choose the popular channels to present, and the unpopular channels have similar performance. The original X-axis is the number of the sorted video files. Since the length of each channel is different, we normalize them to 0–100, and used it to express the entire contents of the channels. The Y-axis is the keep time of the corresponding part of channels. For both applications, the keep time of video files has the tooth-like curve. This strategy seems to be the interval cache which only caches a part of the adjacent

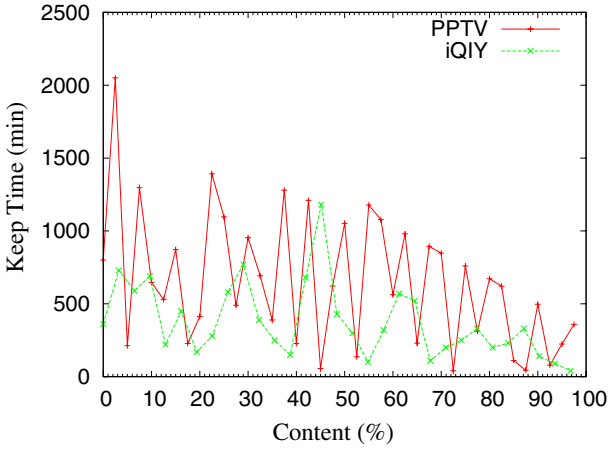

Fig. 2 The caching replacement of PPTV and iQIY

contents. From the discussion about peer collaboration in the previous section, we can find that this caching strategy in PPTV and iQIY is not very good. Therefore, it is urgent that P2P streaming applications need new caching strategy to improve their performance.

3.3.4 Peer Geographic Distribution

We classify the peers communicating with us into three categories according to their IP addresses: peers from China, peers from North America, and users from the rest of the world. To accomplish this, we use whois (Linux command) to query APNIC database [35] to obtain the country information that each peer belongs to with the recorded peer IP address as the input. Figure 3 shows the peer geographic distribution for the four measured channels. Although these peers are not all peers viewing these channels at the collection time, they can also be used to approximately indicate the geographic distribution of PPTV/iQIY peers. We can see that most of peers come from China for all channels. With the decrease of the channel popularity, the audiences are more widely distributed. It is possible that the unpopular channels have to take the poor communication peers, such as overseas peers, as the candidate peers because there are not enough peers to choose. However, in the popular channels, a peer can choose better peers, such as the short distance between them, to communicate from more peers compared to the unpopular channels. We believe that the popular program should have a wider audience distribution, but we only count the peers communicating with our client. From above observation, we know there are still many PPTV/iQIY audiences distributed outside of China. It is difficult to ensure the watching experience of overseas audiences by establishing the dedicated servers. Therefore, for P2P streaming systems, how to effectively use the storage space of each peer to cache data to improve system performance is worthy to study.

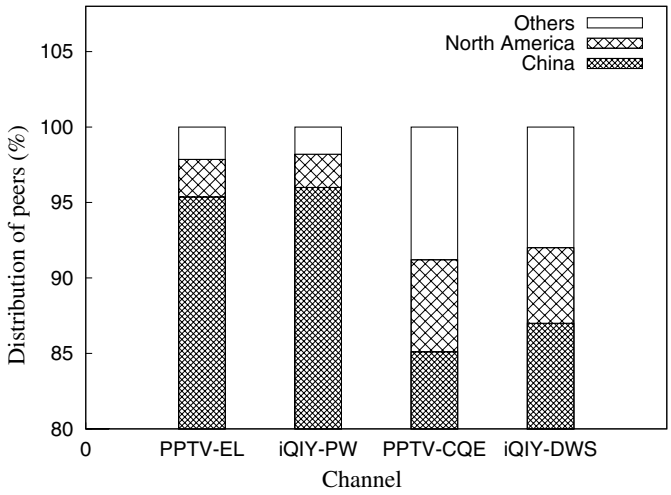

Fig. 3 Peer geographic distribution for different channels

4 Collaborative Caching Mechanism

To improve the caching efficiency of P2P streaming networks, we will introduce a caching strategy in this section.

4.1 Chunk Value

Chunk is the minimum unit of the video data requested by peers. In this paper, we propose the concept of the chunk value (CV) for designing the cache replacement better. If a chunk is never requested, it will be of no value and is not necessary to be kept in the caching space. If a chunk with higher CV, we will keep it longer in the cache because the other peers may be more need to download it. However, to accurately answer which chunk is more valuable is very difficult in a distributed environment. In P2P networks, if a chunk is very popular (that is many peers own it), a peer can quickly gain it from nearby peers. On the contrary, if a chunk is unpopular, it is difficult that a peer gets it, perhaps to go through great distances because only few peers own it. By some inspiration, we use the communication delay as a chunk's CV. The higher the communication delay, the higher the value of the file being accessed. Our basic point is if a file is access by a lot of remote peers, then can be considered it is useful. A chunk c's CV V_c at time t in Eq. 4 is refined to

$$V_c(t+1) = \mu V_c(t) + \eta \sum_{i \in t} d_i(c) \tag{4}$$

where, $0 < \mu, \eta < 1$, $\mu + \eta = 1$. The larger the value of μ, the more important history requested information on the chunk C. On the contrary, the larger the value of η, the more important recent requested information on the chunk C. The value $d_i(c)$ is the delay of the i-th request for chunk C in the period t. If a chunk has more requests or the delay of its request is high, then it will have a higher CV value. The value of the data block is calculated separately by each node, which can be used as an important reference factors in cache replacement mechanism.

4.2 Requesting Distribution Factor

In a distributed environment, each peer only has part of the information of other peers in the network. Therefore, is is difficult to answer which blocks should be replaced when the caching space is full. Our measurement study found that the existing P2P streaming systems deploy a large number of dedicated servers to cache video data in order to improve the user experience. Although this mechanism is simple and feasible, its overhead is huge and scalability is poor. Moreover, it is difficult to cover the entire network, such as for the overseas users. In this paper, we not only take advantage of the chunk value to improve the caching strategy, but also research how to balance the distribution of video data. If a peer caches the data of the peers far away from it as much as possible, the peers closing it will no longer need to request the same data from the distant

peers. For example, the peers in US cache data from the peers in China more, rather than the data from the peers in US. So, if other peers in US hit the caching in the local peer cluster, it will greatly improve download speeds and can also greatly reduce the flow of data across multiple ISPs. Our simulation results show that this heuristic idea is feasible. At the same time, we also consider the factor of requests from other peers. a data block is requested by the local peers frequently, which indicates it is playing a role as a caching relay. Therefore we should encourage this data stream to improve system efficiency. The requested factor of the chunk C F_c is defined at Eq. 5.

$$F_c = \frac{T(C)}{\sum_{i=1}^{n} d_i(C)} \tag{5}$$

where, $T(C)$ is the latency overhead that a peer gets the data block C. The greater $T(C)$ of a data block, the farther the distance from the current peer. $d_i(C)$ is the delay of the ith requested mission for the data block C, and n is the total number of requests from other nodes on the data block C. Based on the previous discussion, the larger F_c of a data block, the farther peers that the data comes from. And there are more local peers to request it. If we keep this data in the caching space, it will be beneficial to distribute the cache more effectively, and bring less transmission delay and communication overhead for the subsequent data requests. Accordingly, the request distribution factor of a data block can also be used as a reference factor for designing the cache replacement strategy.

4.3 Replacement Algorithm

In P2P streaming systems, the played video data will be cached by the caching system that will replace some of the video from it when its space is full. In order to make a decision on replacement better, we define the data block C's replacement factor $R_c(t+1)$ at time t as following,

$$R_c(t+1) = \lambda V_c(t+1) + (1-\lambda)F_c \tag{6}$$

where, λ is an adjustable parameter, and it can be adjusted automatically when the system is running. The initial value of λ is 0.5. For example, if the distribution of peers in the network is more concentrated, we can increase the value of λ to take more consideration on the value of the data block; if the distribution is very fragmented, we can reduce the value of λ to implement a more balanced distribution of the caching.

Each peer is responsible for calculating the value of chunk value and requesting distribution factor for its cached data. When a data block needs to be replaced, the system will compute the replacement factor of all blocks according to the formula 6, and replace the block with the smallest replacement factor. The collaborative cache replacement algorithm is presented in Algorithm 1 in details.

Algorithm 1 The caching replacement algorithm

```
1: while P2P Client running do
2:     put downloaded chunk into the caching;
3:     if the caching is full then
4:         calculate R_c for each chunk C;
5:         evict the cached chunk C with the lowest R_c;
6:     end if
7: end while
```

5 Evaluation

5.1 Simulation Method

In a real environment, P2P overlay network contains tens of thousands peers, even millions. A peer can join the network at any time, and leave the network without any notice as well. Dealing with such a dynamic environment is very challenging. In our experiments, we use the popular PeerSim [36] simulator as the driven kernel. PeerSim has been designed to simulate large-scale P2P networks. It has great scalability and can support thousands of peers. Peersim is supported by the Apache open source project, based on component architecture, it is developed by Java, which allow us to easily implement the protocol prototype by combining different Peersim components. In order to simulate the P2P streaming system, we implement P2P streaming application protocol based Peersim, which can simulate P2P streaming systems. Due to both PPTV and iQIY are not open source systems, their caching algorithms cannot be simulated because we do not know their mechanism. For comparative analysis, we take the data obtained by using measurement experiments to present PPTV and iQIY, and for the proposed scheme, we using the simulation based on Peersim.

Table 3 is our experimental parameters. For the proposed method, we use PeerSim to simulate it. Our simulation network size is 5 thousand peers. We use 1–5 peers as the dedicated servers, and set the cache size of each peer to be 1 G. The simulation is run for 24 h to calculate the cache hit ratio, load and cache replacement feature of each peer. We use measurement studies to analyze PPTV and iQIY. That is to use two computers to run the PPTV/iQIY client, and then collect the computer's communication packets and the client's caching changes. Thus we can estimate the two P2P clients' caching strategy, the cache hit rate and so on. Their experimental cycle is the measurement cycle for the channels. For example, for

Table 3 List of experimental parameters

Experimental type	Proposal Simulation	PPTV Measurement	iQIY Measurement
Experimental cycle (h)	24	32–45	39–63
Network size (K) (the number of peers)	5	10–3000	20–2400
The number of dedicated servers	2	2–10	2–10
The cache size of a peer (G)	1	1	1

iQIY, the experimental cycle of the channel PW is 39 h. The network size of the two clients is also estimated by us. The network size given in Table 3 for PPTV/iQIY is the number of IP that had communicated with the measurement computers. In fact, the actual network size is larger than that of measurement. We counted some IP in measurement studies, and found that they appeared on many channels with a lot of data transmission, especially for the unpopular channels, and long online. We think they are the IP of the dedicated servers in our paper. For PPTV/iQIY, the client cache size is set to 1 G like Proposal.

5.2 Caching Hits

In order to evaluate the cache efficiency for requesting files, we define the cache hits of a peer as the number of downloading files from this peer by other peers. In P2P streaming systems, when a peer downloads data from other peers, it actually needs to access the cached data of these peers. Obviously, the higher the cache hits, the better cache utilization an algorithm can achieve. We run PPTV and iQIY 24 h, and use Wireshark to collect their communicating data. After cleaning and analyzing the collected data, we can obtain the cache hits of PPTV and iQIY. As for our strategy, we get similar information from the simulated experiment.

Figure 4 depicts the cache hits in a peer for three strategies, where the y-axis shows the cumulative number of caching hits. Our strategy (Proposal) has the best performance, and the cached files totally have been accessed 732 times within 24 h. The two real systems have similar performance, and the total number of requesting cached files does not exceed 500. However, the amount of access to a peer in the network is very large. For example, there are about 3M peers communicating with our clients for popular channel EL in PPTV. This is to show that, for a peer, it gets a lot of data from the other peers, but gives them less data. Form the figure, we can know that there is less access in the middle of the night because the curve of cache hits is

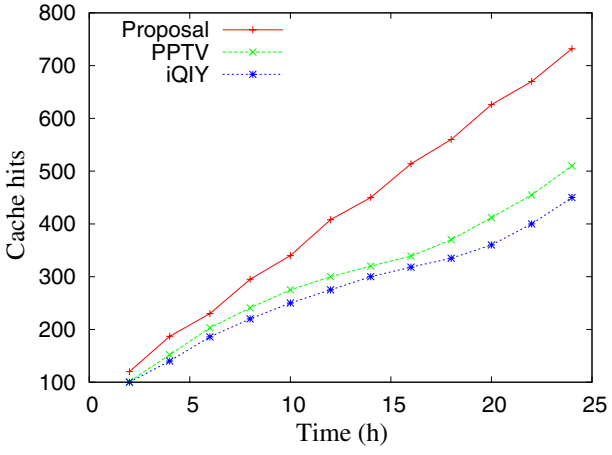

Fig. 4 The caching hits of three strategies in 24 h

flat between 10 and 16. This is because the number of users at midnight is less. Our strategy has not such rule because the data requests are random in the simulation experiment. In a word, we can estimate that these P2P applications have low cache utilization based on our measurement data.

5.3 System Load

Measurement studies cannot get the system load of the whole network. We use simulation experiments to analyze it for PPTV and iQIY in this paper. According to the previous measurement experiment, we can roughly estimate that PPTV and iQIY use alternate caching mechanism (interval cache) to replace the cached data. For simplicity, we consider that the real P2P streaming systems use interval cache as their caching strategies. To analyze our strategy compared to the existing applications, we use Peersim to simulate the system with the algorithm of interval cache and our strategy respectively. In the simulation experiments, we set two peers as dedicated servers, and collect the communication traffic among peers under different the network sizes. As can be seen from Fig. 5, our strategy can reduce the traffic load of the dedicated server compared to the systems with interval cache, where the traffic messages are collected in one simulation cycle. The proposed algorithm can reduce about 38% of the traffic when the network size is 5000 peers.

5.4 Peer Cooperation

In order to compare PPTV/iQIY with our strategy for peer cooperation, we designed an experiment that calculates the ratio of data sharing. For PPTV and iQIY, the experimental results are generated by measuring the actual system experiments. To make comparison simpler and clearer, we only choose the popular channels. As it can be

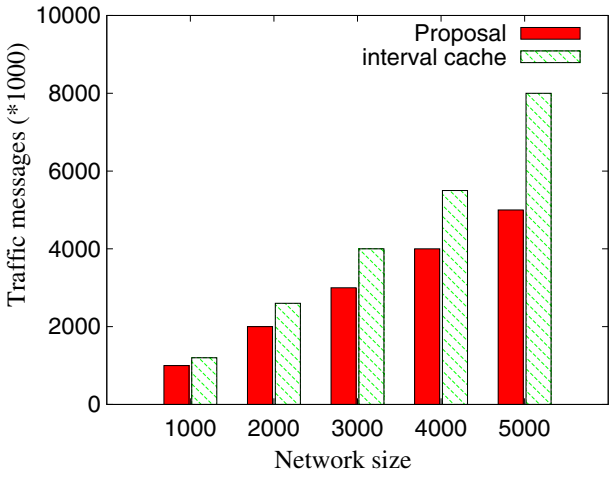

Fig. 5 The system load for different strategies

seen from Fig. 6, the proposed algorithm has, compared to PPTV and iQIY, greater improvement. On average, a peer in simulation sends 12% of all sent data to top 100 peers, 60% to top 500 peers, and approximately 70% to all peers who also provide data to it. These data show that the proposed algorithm has good collaborative nodes. In other word, the cache data of each peer has been utilized better in our strategy. However, the actual system there is poor performance, such as PPTV, only 8% of two-way communication traffic, which can be considered as the one-way flow of data that is a waste of bandwidth.

5.5 Confidence Interval

In the above, we evaluated our algorithm from three metrics: caching hits, system load and peer cooperation. To evaluate the reliability of these experimental results, we run our experiments one hundred times to calculate their confidence intervals. In order to make it easier to calculate, we adjust these metrics. For the caching hits in the Sect. 5.2, we change it to the average caching hit rate. For a peer, its caching hit rate is the ratio of the number of caching hits for it to the total number of access for the peer, where the total number of access is considered to be the number of peers that have data communication with the peer. The average caching hit rate is the average value of caching hit rates of all peers in the simulations. For system load (the total number of the traffic messages of the system), we change it to the average number of the traffic messages of each peer, which is called the peer average traffic messages. For peer cooperation, the cooperative rate of a peer (CR) is defined as

$$CR = \frac{N_{mutual}}{N_{all}} \tag{7}$$

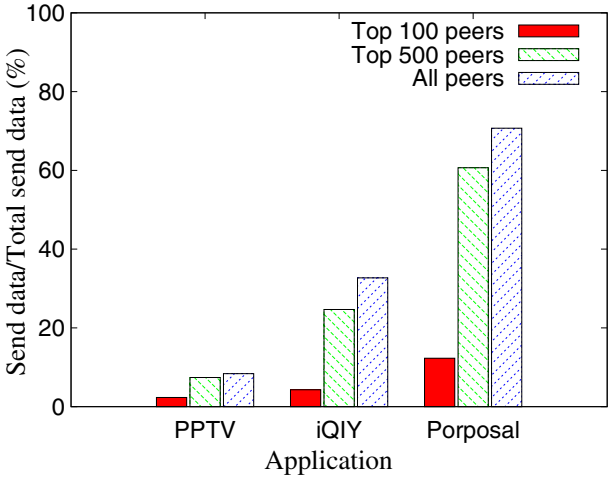

Fig. 6 The peer collaborative

Table 4 List of the experiment results

Metric	Algorithm	Value
Average caching hit rate	Proposal	30%
	PPTV	4%
	iQIY	3.8%
Peer traffic messages	Proposal	1000
	Interval cache	1600
Cooperative rate	Proposal	70.7
	PPTV	8.3
	iQIY	32.7

Table 5 List of the experimental parameters for sampling

No.	Average caching hit rate		Peer traffic messages		Cooperative rate	
	Peers	Servers	Peers	Servers	Peers	Servers
1	5000	5	5000	5	5000	5
2	5000	4	5000	4	5000	4
3	5000	3	4000	4	5000	3
4	5000	2	4000	3	4000	4
5	5000	1	3000	3	4000	3
6	4000	5	3000	2	4000	2
7	4000	3	2000	2	3000	3
8	3000	3	2000	1	3000	2
9	2000	2	1000	2	2000	2
10	1000	1	1000	1	1000	1

where, N_{mutual} is the data sent by a peer to all peers that provide data for it, and N_{all} is the total data sent by this peer. Based on the new metrics, we converted the above experimental results (Sects. 5.2–5.4) into Table 4. The caching hit rate of our algorithm is much higher than that of PPTV and iQiy, reaching 30%. As for the system load, the algorithm of interval cache (we estimated that PPTV and iQIY use this cache algorithm in Sect. 3) is 1600, and ours is 1000. For peer cooperation, our algorithm is about 70%, iQiy is 32%, and PPTV is only 8%.

We use different parameters to run simulation experiments. These parameters are the number of peers and the number of dedicated servers, in which the number of peers changes from 1000 to 5000 and the number of dedicated servers is 1–5. We run 100 experiments and keep 40 of that to calculate the conference interval for three evaluation metrics. Table 5 lists the experimental parameters for the 10 sampling for each metric, that is, the number of peers and the number of dedicated servers used in Peersim. We calculate their confidence intervals with a confidence level of 95%, and the result is shown in Fig. 7. We can see that the confidence interval of average caching hit rate is [29.3% 30.6%] for our algorithm. The confidence interval

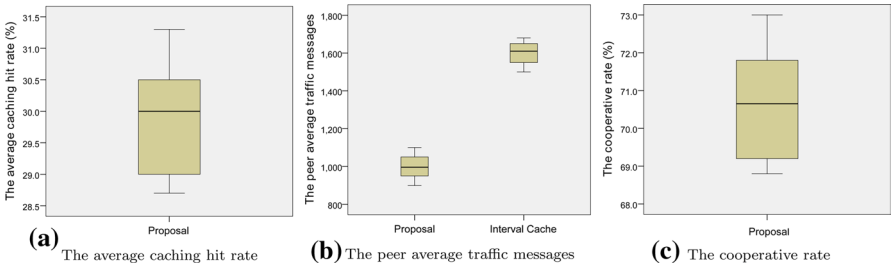

Fig. 7 The confidence interval for three evaluation metrics

of the peer traffic message of Proposal is [953 1047], and that of Interval Cache is [1556 1644]. As for the cooperative rate, its confidence interval for our algorithm is [69.6% 71.6%]. These data indicate that our experimental results are reliable.

6 Conclusion

Caching is an effective way not only to relieve the pressure on Internet communication, but also to improve the user experience for P2P streaming systems. This paper presents a relatively new cooperative caching mechanism to improve the performance of P2P streaming systems and reduce the traffic brought to the Internet by P2P streaming networks. In order to get better caching strategy, we measured the existing large-scale P2P streaming system PPTV and iQIY, and found some valuable features and problems. For example, most P2P streaming systems, such as PPTV and iQIY, use the dedicated servers to improve the user experience. This strategy not only has centralized inherently defects, but also there is a huge investment in hardware spending. Based on the analysis of measured data, we proposed two concepts: chunk value and requesting distribution factor. In simple terms, the chunk value is the frequency of a chunk to be accessed, and the requesting distribution factor is the cumulating delay of getting a data block. We use the replacement factor to help P2P streaming systems decide which cache should be replaced. The replacement factor is generated by the following rules: the larger chunk value and requesting distribution factor, the greater replacement factor for a chunk. The core purpose of the proposed strategy is to keep as much as possible those cached video data that not only has high frequency, but also can ensure the balanced distribution. The simulation results show that, compared to the existing systems, the proposed method has a better performance on multiple metrics, such as cache hits, system load and peer cooperation.

The popular is constantly changing. An object is popular at the current time, but it maybe become unpopular in next time. Our future work is to study the prediction mechanism of the object popularity. To further improve the efficiency of the system cache, we also plan to clean and reorganize measurement data, and implement the simulation experiment based on trace-driven to enhance the reliability of the experimental results.

Acknowledgements This work is supported by the National Key Research and Development Program of China under Grants 2016QY01W0202 and 2016YFB0800402, National Natural Science Foundation of China under Grants 61572221, U1401258, 3761433006, 61502185 and 61173170, Major Projects of the National Social Science Foundation under Grant 16ZDA092, Science and Technology Support Program of Hubei Province under Grant 2015AAA013, Science and Technology Program of Guangdong Province under Grant 2014B010111007, Guangxi High level innovation Team in Higher Education Institutions Innovation Team of ASEAN Digital Cloud Big Data Security and Mining Technology, and Excellent Young and Middle-aged Science and Technology Innovation Team Plan of Hubei Higher Education (T201807).

References

1. eMarketer: Worldwide internet and mobile users: emarketer's updated estimates and forecast for 2017 to 2021. https://www.emarketer.com/report/worldwide-internet-mobile-users-emarketers-updated-estimates-forecast-20172021/2002147. Accessed 26 Nov 2018
2. Cisco Inc.: Cisco visual networking index: forecast and methodology, 2016–2021. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html. Accessed 26 Nov 2018
3. Papapanagiotou, I., Nahum, E.M., Pappas, V.: Smartphones vs. laptops: comparing web browsing behavior and the implications for caching. In: ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems, SIGMETRICS '12, London, UK, 11–15 June 2012, pp. 423–424 (2012)
4. Alexander, H., Khalil, I., Cameron, C., Tari, Z., Zomaya, A.Y.: Cooperative web caching using dynamic interest-tagged filtered bloom filters. IEEE Trans. Parallel Distrib. Syst. **26**(11), 2956–2969 (2015)
5. Hasslinger, G., Ntougias, K., Hasslinger, F., Hohlfeld, O.: Performance evaluation for new web caching strategies combining LRU with score based object selection. Comput. Netw. **125**, 172–186 (2017)
6. Levin, D., LaCurts, K., Spring, N., Bhattacharjee, B.: Bittorrent is an auction: analyzing and improving bittorrent's incentives. In: Proceedings of the ACM SIGCOMM 2008 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Seattle, WA, USA, 17–22 Aug 2008, pp. 243–254 (2008)
7. Kumar, N., Zeadally, S., Rodrigues, J.J.P.C.: QoS-aware hierarchical web caching scheme for online video streaming applications in internet-based vehicular ad hoc networks. IEEE Trans. Ind. Electron. **62**(12), 7892–7900 (2015)
8. PPTV: http://www.pptv.com. Accessed 26 Nov 2018
9. iQIY: http://www.iqiyi.com. Accessed 26 Nov 2018
10. QQLive: http://v.qq.com. Accessed 26 Nov 2018
11. UUSee: http://www.uusee.com. Accessed 26 Nov 2018
12. SopCast: http://www.sopcast.com. Accessed 26 Nov 2018
13. da Silva, A.P.C., Leonardi, E., Mellia, M., Meo, M.: Chunk distribution in mesh-based large-scale P2P streaming systems: a fluid approach. IEEE Trans. Parallel Distrib. Syst. **22**(3), 451–463 (2011)
14. Zhao, J., Zhang, P., Cao, G., Das, C.R.: Cooperative caching in wireless P2P networks: design, implementation, and evaluation. IEEE Trans. Parallel Distrib. Syst. **21**(2), 229–241 (2010)
15. Pacifici, V., Lehrieder, F., Dán, G.: Cache bandwidth allocation for P2P file-sharing systems tominimize inter-ISP traffic. IEEE/ACM Trans. Netw. **24**(1), 437–448 (2016)
16. Das, S.K., Naor, Z., Raj, M.: Popularity-based cachingfor IPTV services over P2P networks. Peer-to-Peer Netw. Appl. **10**(1), 156–169 (2017)
17. Wierzbicki, A., Leibowitz, N., Ripeanu, M., Wozniak, R.: Cache replacement policies revisited: the case of P2P traffic. In: 4th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2004), 19–22 April 2004, Chicago, IL, USA, pp. 182–189 (2004)
18. Hefeeda, M., Hsu, C.-H., Mokhtarian, K.: Design andevaluation of a proxy cache for peer-to-peer traffic. IEEE Trans. Comput. **60**(7), 964–977 (2011)

19. Zhu, Q., Lee, D.L., Lee, W.-C.: Collaborative caching for spatial queries in mobile P2P networks. In: Proceedings of the 27th International Conference on Data Engineering, ICDE 2011, 11–16 April 2011, Hannover, Germany, pp. 279–290 (2011)
20. Kim, J., Im, H., Bahk, S.: Adaptive peer caching for P2P video-on-demand streaming. In: Proceedings of the Global Communications Conference, 2010. GLOBECOM 2010, 6–10 Dec 2010, Miami, FL, USA, pp. 1–5 (2010)
21. Dai, J., Li, B., Liu, F., Li, B., Jin, H.: On the efficiency of collaborative caching in ISP-aware P2P networks. In: INFOCOM 2011. 30th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 10–15 April 2011, Shanghai, China, pp. 1224–1232 (2011)
22. Silverston, T., Jakab, L., Cabellos-Aparicio, A., Fourmaux, O., Salamatian, K., Cho, K.: Large-scale measurement experiments of P2P-TV systems insights on fairness and locality. Signal Proc. Image Commun. **26**(7), 327–338 (2011)
23. Brienza, S., Cebeci, S.E., Masoumzadeh, S.S., Hlavacs, H., Özkasap, Ö., Anastasi, G.: A survey on energy efficiency in P2P systems: file distribution, content streaming, and epidemics. ACM Comput. Surv. **48**(3), 36 (2016)
24. Liu, Y., Guo, L., Li, F., Chen, S.: A case study of traffic locality in internet P2P live streaming systems. In: 29th IEEE International Conference on Distributed Computing Systems (ICDCS 2009), 22–26 June 2009, Montreal, QC, Canada, pp. 423–432 (2009)
25. Seibert, J., Zage, D., Fahmy, S., Nita-Rotaru, C.: Experimental comparison of peer-to-peer streaming overlays: an application perspective. In: LCN 2008, The 33rd IEEE Conference on Local Computer Networks, The Conference on Leading Edge and Practical Computer Networking, Hyatt Regency Montreal, Montreal, QC, Canada, 14–17 Oct 2008, Proceedings, pp. 20–27 (2008)
26. deploying PlanetLab: An open platform for developing and accessing planetary-scale services. http://www.planet-lab.org. Accessed 26 Nov 2018
27. Pai, V.S., Kumar, K., Tamilmani, K., Sambamurthy, V., Mohr, A.E.: Chainsaw: eliminating trees from overlay multicast. In: Peer-to-Peer Systems IV, 4th International Workshop, IPTPS 2005, Ithaca, NY, USA, 24–25 Feb 2005, Revised Selected Papers, pp. 127–140 (2005)
28. Castro, M., Druschel, P., Kermarrec, A.-M., Nandi, A., Rowstron, A.I.T., Singh, A.: Splitstream: high-bandwidth multicast in cooperative environments. In: Proceedings of the 19th ACM Symposium on Operating Systems Principles 2003, SOSP 2003, Bolton Landing, NY, USA, 19–22 Oct 2003, pp. 298–313 (2003)
29. Mu, M., Ishmael, J., Knowles, W., Rouncefield, M., Race, N.J.P., Stuart, M., Wright, G.: P2P-based IPTV services: design, deployment, and qoE measurement. IEEE Trans. Multimed. **14**(6), 1515–1527 (2012)
30. Li, B., Ma, M., Jin, Z., Zhao, D.: Investigation of a large-scale P2P voD overlay network by measurements. Peer-to-Peer Netw. Appl. **5**(4), 398–411 (2012)
31. Ciullo, D., Mellia, M., Meo, M., Leonardi, E.: Understanding P2P-TV systems through real measurements. In: Proceedings of the Global Communications Conference, 2008. GLOBECOM 2008, New Orleans, LA, USA, 30 Nov–4 Dec 2008, pp. 2297–2302 (2008)
32. Joost: http://www.joost.com. Accessed 26 Nov 2018
33. WireShark: http://www.wireshark.com. Accessed 26 Nov 2018
34. Tcpdump: http://www.tcpdump.org. Accessed 26 Nov 2018
35. APNIC: http://www.apnic.net. Accessed 26 Nov 2018
36. Montresor, A., Jelasity, M.: Peersim: a scalable P2P simulator. In: Proceedings P2P 2009, Ninth International Conference on Peer-to-Peer Computing, 9–11 Sept 2009, Seattle, WA, USA, pp. 99–100 (2009)

Guoqiang Gao received the M.S. and Ph.D. in Computer Science from Huazhong University of Science and Technology, China in 2007 and 2011 respectively. He is currently an associate professor in School of

Media and Communication at Wuhan Textile University, China. His research interests include peer-to-peer computing, cloud computing, and data mining. He is a member of IEEE.

Ruixuan Li received the B.S., M.S., and Ph.D. degrees from School of Computer Science and Technology at Huazhong University of Science and Technology in 1997, 2000 and 2004 respectively. He is currently a professor of School of Computer Science and Technology at Huazhong University of Science and Technology, and is the director of the Intelligent and Distributed Computing Laboratory. His research interests include big data, cloud computing, and system security. He is a member of IEEE and ACM.