Content Harvest Network: Optimizing First Mile for Crowdsourced Live Streaming

Haitian Pang, *Student Member, IEEE*, Zhi Wang, *Member, IEEE*, Chen Yan, Qinghua Ding, Kun Yi, Jiangchuan Liu, *Fellow, IEEE*, and Lifeng Sun, *Member, IEEE*

***Abstract*— Crowdsourced live streaming (CLS), such as Twitch.tv and Inke.tv, has emerged as an important multimedia application in recent years. Video delivery in such CLS service involves two steps: 1) video uploading—video streaming (i.e., a live channel) generated from a broadcaster is uploaded to the server, which we call the "first mile" network and 2) video distribution—the video streaming is then delivered to viewers in the channel. Today's CLS services usually use conventional content delivery network solutions to address the video distribution problem, while little attention has been paid to improve the video uploading quality. Our measurement study shows that the first mile network causes 17% viewer rebuffers, and some viewers quit the channel once encountering rebuffer. In this paper, we propose a content harvest network (CHN) architecture to address the uploading problem in the CLS service. Specifically, the CHN architecture employs edge devices in the network as relays to receive the streaming uploaded by broadcasters and then forward to the central servers. On one hand, we need to reduce the latency since it is live streaming; on the other hand, we must provide sustainable upload bandwidth. It is challenging to achieve both at the same time, especially in such high dynamic system as CLS. In order to provide global optimal and real-time assignment, we propose a hybrid solution, i.e., centralized and distributed assignment. Specifically, we formulate the centralized relay assignment problem as an optimization problem to achieve both low latency and sustainable bandwidth. To cope with the frequent channel establishments, we use a multi-armed bandit method to characterize the time-variant network condition. Experiment results on a large-scale trace provided by Inke.tv show that our solution can reduce the overall viewer cost by 40% compared to state-of-the-art solutions. The viewers' rebuffer can also be reduced by 50%.**

Manuscript received March 15, 2018; revised July 15, 2018; accepted July 20, 2018. Date of publication August 2, 2018; date of current version July 1, 2019. The work of H. Pang, C. Yan, Q. Ding, K. Yi, and L. Sun was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61472204 and Grant 61521002, in part by the Beijing Key Laboratory of Networked Multimedia under Grant Z161100005016051, and in part by the Key Research and Development Project under Grant 2018YFB1003703. The work of Z. Wang was supported by NSFC under Grant 61531006. The work of J. Liu was supported in part by the Canada NSERC Discovery Grant, and in part by NSERC E.W.R. Steacie Memorial Fellowship. This paper was recommended by Associate Editor Y. Wen. *(Corresponding author: Lifeng Sun.)*

H. Pang, C. Yan, Q. Ding, K. Yi, and L. Sun are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: pht14@mails.tsinghua.edu.cn; yc.1995@foxmail.com; dqh14@qq.com; james1996thu@163.com; sunlf@tsinghua.edu.cn).

Z. Wang is with the Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055, China (e-mail: wangzhi@sz.tsinghua.edu.cn).

J. Liu is with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: jcliu@cs.sfu.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2018.2862619

***Index Terms*— Crowdsourced live streaming, broadcaster uploading, content harvest network.**

I. Introduction

The current data traffic on the Internet is dominated by video streaming, which will account for 82% by 2021. Live streaming is one of the most important fuel for the video burst, which is forecasted to grow 15-fold from 2016 to 2021, and account for 13% of Internet video traffic by 2021, according to Cisco's report [1]. The newly emerged crowdsourced live streaming (CLS) is gaining rapid increase. such applications as Facebook Live, Inke.tv, and Twitch.tv have attracted millions of daily active users. The key idea of CLS is that numerous widely-distributed broadcasters provide live streaming to viewers using mobile computing devices (e.g., smartphone and tablet). As a practical example, Inke.tv [2], one of the largest CLS platform in China, allows anyone to broadcast his content to massive viewers via mobile devices. The monthly active user (MAU) of Inke.tv reached 24.6 million, and the registered member reached 164.6 million by March 2017. End users in CLS not only consume contents but also produce contents. One broadcaster can establish only one *channel* concurrently, but he can establish different channels in different time periods. The lifecycle of one viewer watching one channel is defined as a *session*.

Different from Video on Demand (VoD) services (e.g., Hulu or Netflix) and professional live streaming services (e.g., ESPN), the CLS service faces an especially severe challenge for low latency and sustainable bandwidth as the most broadcasters employ ordinary mobile devices and unstable network for live streaming. Specifically, the challenges of video delivery in CLS are provided as follows:

- The broadcaster's bandwidth of first-mile upload network is much smaller than the download link due to the asymmetry of today's Internet architecture.
- In order to provide real-time interaction between broadcasters and viewers in the CLS service, ultra-low network transmission latency is required in the unstable network.
- The massive broadcasters establish and terminate broadcast channels arbitrarily, incurring high churn rate of the streaming service. In our measurement, the channel number in peak hours is 1.87× of in low hours.

Due to the above challenges, a novel architecture for video uploading is expected to support the massive and unstable uploading sources in the CLS service. We conduct a measurement study on Inke.tv, which is one of the most popular CLS provider in China supported by the conventional CDN [3], and find the first-mile network accounts for 17% viewer rebuffers. Moreover, the larger broadcaster uploading latency causes more viewer rebuffers, and the viewers may quit a channel when encountering rebuffer. We further observe that the streaming service is vulnerable and sensitive when the broadcaster's uploading capacity varies frequently. Fortunately, the measurement results show that employing relay network can potentially improve the uploading network performance and further cope with the above challenges in CLS [4].

Inspired by the measurement results and the trend that edge computing [5] and smart router [6] are playing an important role in content delivery, we design a relay-based content harvest network to address the first mile network problem for the CLS service. The key idea is that using the relay technique, we can potentially find an alternative network path with higher available bandwidth, lower latency, and lower loss rate. Specifically, CHN utilizes dedicated edge nodes to form a relay network, providing the broadcasters with optional routing schemes with better network condition. In this way, live streaming can be delivered directly to the server or relayed to the relays first and then delivered to the server. The challenges of this novel architecture are as follows:

- How to choose the right edge devices to achieve both low latency and sustainable bandwidth in the first mile network for a broadcast channel.
- How to dynamically adjust the relay assignment as the network conditions vary over time and broadcast channels establish and terminate frequently.
- How to allocate the resource of the relay network efficiently among the broadcast channels to minimize the overall cost.

To address the above challenges in CHN, we propose a hybrid (centralized and distributed) solution to assign the relays. Specifically, we formulate the centralized relay assignment as an optimization problem, in which we achieve the multi-objective optimization, i.e., low latency, low loss rate, and sustainable bandwidth. Then we design an approximated algorithm using rounding technique and further design a fast implementation in polynomial time. As the network condition varies over time, we need a dynamic relay assignment strategy to capture the real-time network condition. In this way, we enable the real-time decision in broadcaster devices using the MAB-based method [7], which models an agent with historical information taking action repeatedly to maximize the accumulated reward. In our scenario, it can *exploit* the current optimal relay and *explore* the potential better relays in the time-variant network condition. Experimental results on a large scale dataset of Inke.tv show that with the aid of CHN, the total viewer cost is reduced by 40% compared to the conventional CDN-based method. The MAB-based method can reduce the latency by 27% and improve the bandwidth by 140% compared to the static assignment. Viewers' rebuffer can be reduced by 50% in peak hours using the proposed method.

The remainder of this paper is organized as follows. Section II introduce the insight of the real-world data measurement. Section III introduce the architecture design. Section IV present the centralized relay assignment method. Section V present the distributed relay assignment method. Experiment results are provided in Section VI. We introduce the related work in Section VII and conclude in Section VIII.

TABLE I
FOUR DATASETS USED IN THE MEASUREMENT

Channel Data	Upload Log	Viewer Data	Download Log
broadcaster ID	channel ID	viewer ID	channel ID
channel ID	packet ID	channel ID	packet ID
establish time	timestamp	entering time	timestamp
terminate time	latency	exiting time	rebuffering
location (GPS)		location	

II. Data Analysis and Insights

In this section, we investigate the dataset to know how the first mile upload network affects the viewer experience, the challenges in improving the CLS service, and the insights to design the CHN.

A. Dataset Description

Inke.tv is a pioneering CLS provider in China, which allows common users to broadcast live streaming to other mobile users. In order to better understand the inherent challenges and insights of CLS service, we conduct data analysis on the real-trace dataset provided by Inke.tv from December 9, 2016 to February 27, 2017, containing 1.2M broadcasters establishing 25M channels within 80 days, bringing about 1.5B views. Note that one channel means one broadcaster establishing one streaming session. We believe that this dataset is representative of the CLS service, as the time span is long enough, the involved sessions are large in number. In addition, as the dataset is provided directly by a CLS service provider, many items like the location of broadcasters and viewers are also included.

Table I shows the datasets we use to perform the measurement. Each broadcaster entry corresponds to one established channel, which contains the broadcaster ID, the channel ID, the timestamp when the channel is established and terminated, and the broadcaster location (recorded by GPS, in longitude and latitude). We also collect the upload streaming log, which contains the channel ID, the packet ID, the timestamp when the packet is sent, and the corresponding uploading latency. Each viewer entry corresponds to one view session in a broadcast channel, containing the viewer ID, the channel ID, the timestamp when viewers entering and exiting, and the viewer location (in longitude and latitude). We also collect the network log of packet-level viewer entry, in which each entry contains the channel ID, the packet ID, the timestamp, and the viewer rebuffering (yes or no). Joint utilizing the upload streaming log and the viewer network log, we can investigate the impact of the uploading network on viewers.

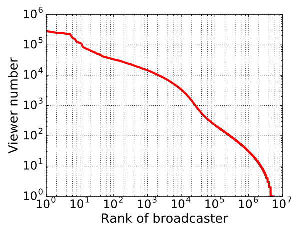


Fig. 1. Broadcasters' viewer number v.s. viewers in descent order of viewer number.

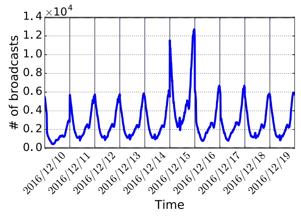


Fig. 2. Broadcaster number over time.

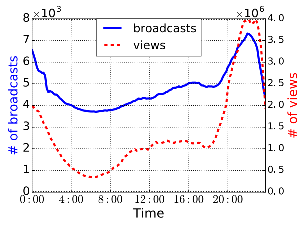


Fig. 3. Broadcaster and Viewer number in a day.

B. Overview of the CLS Service

In order to obtain the popularity pattern, we investigate the average viewer number of a broadcaster based on his historical sessions. The intuition is straightforward: the popularity of a broadcaster is stable, as each broadcaster has his own fans. Fig. 1 shows that the distribution of the broadcaster's average viewer number is highly skewed. The observation inspires us to estimate the popularity of a channel considering instant viewer number and the broadcaster's historical popularity. We further prioritize the channel based on the estimated popularity as optimizing the more popular channels will benefit more viewers.

The curve in Fig. 2 represent the broadcaster number in ten days. The broadcaster numbers over time bear periodic pattern. Then, we plot the broadcaster and viewer number in the different time periods in one day on one CDN node in Fig. 3, and notice that the peak hour of broadcasters and viewers are both around 22:00. The broadcaster number can reach 7,000 in the peak hour. The broadcaster number in the whole system is even more. A large number of broadcasters make CLS a multi-source service, which is computation-consuming and calls for high scalability. We further investigate the pattern of the broadcaster channel establishment and termination in Fig. 4, where the unit time is set to 1 minute, and we find that many channels are established and closed in unit time. In particular, the maximum number of establishment

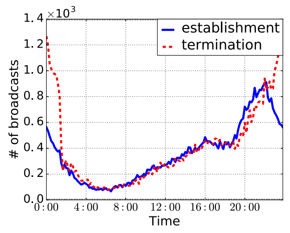


Fig. 4. Channel establishment and termination number.

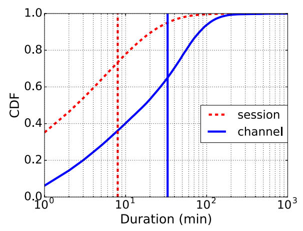


Fig. 5. CDFs of channel duration and session duration.

reaches 900 and appears around 22:00, which is the start time of the peak period in the system. The maximum number of termination reaches 1,200 and appears around 24:00, which is the end time of the peak period.

Fig. 5 shows the duration distribution of channels and sessions. Recall that the channel is defined as the lifecycle of one broadcaster establishing live streaming, and the session is defined as the lifecycle of one viewer watching a channel for one time. The average duration of a channel is 32 minutes, and the average duration of a session is 8 minutes. Both channels and sessions establish in relatively short duration, causing high temporal dynamic of the CLS service. This observation indicates that the channels are highly dynamic, which inspires us to employ real-time assignment to perform fast response.

C. Impact of First Mile Network on Viewer Experience

[8] showed that the viewer rebuffer may either be caused by the first mile (broadcaster's source) or the download network (viewer's side) congestion, we need to first identify the viewer rebuffers caused by the first mile network congestion. Once the first mile network is congested, the CDN server fails to receive the uploading streaming, causing all the viewers to encounter rebuffer, as no buffer is in viewers' devices. While the download network congestion of one viewer cannot affect others. In this way, only part of the viewers will encounter rebuffer. We define the viewer rebuffer percentage as the number of rebuffered viewers divided by the number of all viewers in the same channel in one time slot. Fig. 6 illustrates the distribution of the viewer rebuffer percentage, where the x-axis is the rebuffer percentage, and the y-axis is the percentage of the case that the rebuffer percentage is no larger than x. We find the CDF is a piecewise function, where the case that 100% viewers encounter rebuffer concurrently accounts for 17% total rebuffers, which is caused by the first mile network congestion. This indicates that the file-mile network causes a significant portion of viewer rebuffers.

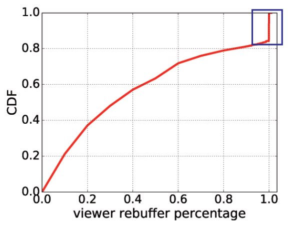


Fig. 6. Distribution of concurrent viewers rebuffer percentage in a channel.

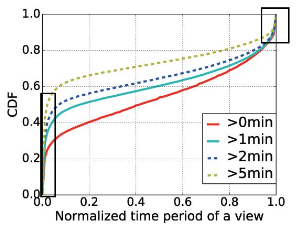


Fig. 7. Fraction of time when viewer rebuffers happen in the view period.

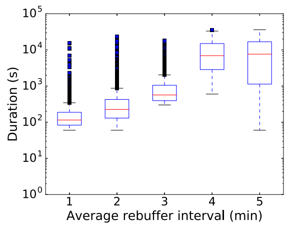


Fig. 8. Session (larger than 1min) duration versus average rebuffer interval.

We further focus on how the viewers react to rebuffers. Fig. 7 depicts the time when viewer rebuffer happens. The x-axis is the time percentage of the viewing lifecycle, which is normalized to [0, 1], and the y-axis is the CDF of rebuffers. We filter the viewing sessions by the duration of viewing above a certain threshold to skip the ultra-short sessions. Specifically, we present the results where the thresholds are set as 0, 1, 2, 5 minutes, respectively. We have two key observations as follows: The first observation is that viewers tend to encounter rebuffer before they exit the channel, i.e., when x approximates 1. We may infer that some viewers exit the broadcast channel because they cannot bear the rebuffers. In this way, it is important to reduce the viewer rebuffer to improve their experience. The second observation is that many viewers encounter rebuffer when they enter the channel, i.e., when x approximates 0. This is caused by long startup delay, i.e., the viewer starts to request streaming from the server, while the streaming has not reached the viewer's device due to the delay between the broadcaster and the user device.

Next, we focus on the relationship between the viewer session duration and the average rebuffer interval in Fig. 8. The x-axis is the average rebuffer interval, which is defined as $\frac{\text{session duration}}{\text{number of rebuffer}}$, and the y-axis is the session duration. Average rebuffer interval can reflect the viewer's watching experience within a channel. We divide the average rebuffer interval into several levels, and a larger level indicates a lower rebuffer frequency. We notice that viewers with lower rebuffer

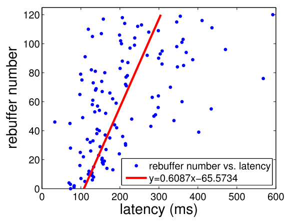


Fig. 9. Correlation between uploading latency and number of viewer rebuffer.

frequency will watch the channel longer. This is due to the fact that frequent rebuffer reduces the viewer QoE and result in viewers quit the channel. Above observations prove that poor first mile network conditions may cause viewers to encounter rebuffer frequently, and this will further cause the viewers to quit the channel.

Knowing that the first mile network will cause viewer rebuffer, we want to further derive the correlation between the upload latency and the viewer rebuffer. In Fig. 9, each sample illustrates the number of viewer rebuffer versus uploading latency. We observe a large uploading latency generally results in more rebuffer number. Specifically, we find the Pearson correlation coefficient equals 0.55, which implies a strong correlation between uploading latency and rebuffer number. We further derive the corresponding linear function

$$y = 0.6087x - 65.5734. \tag{1}$$

This observation motivates us to reduce the uploading latency for potentially better viewer experience.

D. Potential Improvement of Relay Network

We investigate the one-hop relay network based on a *traceroute* dataset collected on the Youku's peer video CDN [9] in December 2015. We analyze the direct latency and the relayed latency between 6365 node pairs. The average number of relays for each node pair is 14.09, and 75% node pairs can improve the network performance via the relay paths. This validates that relay nodes can be utilized to improve the network performance. The challenge is to find the relay which can reduce the latency most significantly. Note that in the cases when the relays cannot improve the network performance, the direct path should be utilized instead of the relayed path.

Furthermore, we look into the dynamic in the relay network. In Fig. 10, each sample is the network latency versus the network bandwidth of one connection between the broadcaster and the relay. The network latency and bandwidth are measured and collected by the *tcpdump* tool. The user establishes the connection with 4 relays, and we notice that even for the same user-relay pair, the network condition varies greatly. The rationale behind this observation is as follows: (1) relays are physically deployed at different locations and with different ISPs, making average network metrics different. (2) Network metrics between a broadcaster and a relay fluctuate with time. This inspires us to dynamically select the optimal relays for broadcasters based on the historical and instant network metrics.

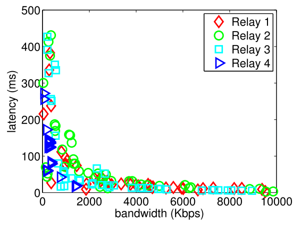


Fig. 10. The bandwidth and latency of the Source-Destination pair using different relays.

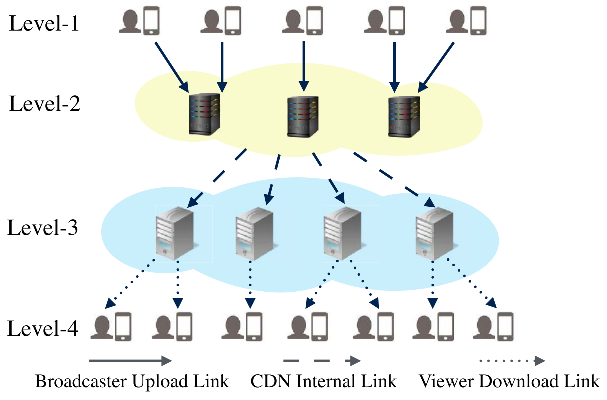


Fig. 11. CDN-based live streaming system.

III. Architecture Design

A. Overview of CDN-Based CLS

We first introduce a typical CDN-based CLS architecture in Fig. 11, which is adopted by many CLS providers like Inke and Twitch. In level-1, the widely distributed broadcasters upload video streaming to the upload server (upServer) of the CDN (level-2). These upServers are geo-distributed so that broadcasters can choose a cost-efficient server for video distribution. Furthermore, the upServer will transcode the video to multiple quality versions, which is computation-intensive. Then the transcoded streaming will be delivered to the download servers (level-3). The download servers are distributed in different geographical locations (e.g., U.S. West and China East), and will serve the viewer requests (level-4) in its region.

Although the network congestion in each link from level-1 to level-4 may affect the viewer QoS, the first mile network between level-1 and level-2 is most critical in the whole path of CLS [8] for the following reasons: first, a single broadcaster uploading link may affect all the viewers in the channel, thus improving the single first mile network can benefit tens of thousands of viewers; second, in most cases the network uploading condition is poor due to limited uploading ability. Therefore, improving the performance of the first mile network is an economical way to benefit all viewers in the channel by only optimizing the single first mile network. To this end, we aim to improve the first mile network using the relay network.

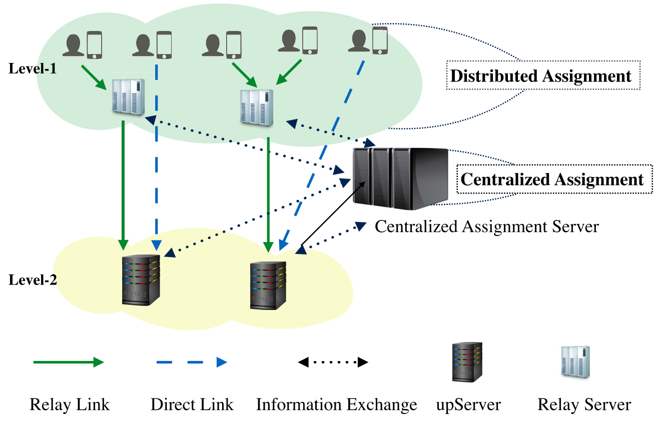


Fig. 12. System overview of content harvest network.

B. Introduction to Content Harvest Network

We present the architecture of CHN in Fig. 12, where we incorporate relays into the first mile network. Each upload streaming can take either the "direct path" (blue arrow) or the "relayed path" (green arrow). When taking a "relayed path", the streaming is first forwarded to a relay, and then forwarded to an upServer. The relays are provided by some edge network operator, who crowdsource the resource of edge devices [5], [6]. When a "relayed path" outperforms the default path, we can choose the "relayed path" to reduce latency and obtain sustainable bandwidth [10].

The relay assignment problem is based on the network performance. Specifically, the network condition between each relay-upServer pair, broadcaster-relay pair and broadcaster-upServer pair is measured and transferred to the centralized assignment server. Keeping track of the network performance between the relay and upServer is feasible, as both are controlled by the CLS provider. However, the network attributes of broadcasters (e.g., IP address, AS) are highly dynamic and large in number, making direct measurement infeasible. Previous works [11], [12] employ data-driven approaches such as cluster methods to predict the network performance. This is a well-studied topic and is out of scope in this work. With the network performance collection from direct measurement and prediction, the network performance keeps up to date, which can be used for relay assignment. Note that the overhead of network measurement is negligible, as all the data is collected in a *passive* way [13], i.e., when a streaming session is established, the network performance is recorded by the relay nodes and upServers, and reported to the centralized assignment server. The update frequency can be set in seconds for accurate monitoring.

Given the system architecture and the insights in the measurement, we present the design principles as follows:

- Hybrid Solution: We implement centralized assignment to optimize the multi-objective relay assignment problem of all the broadcasters periodically. In addition, the distributed assignment is required to cope with the relay assignment problem in real-time when one specific broadcast channel establishes.

- Prioritized broadcast channel: Different channels serve different number of viewers, and the popular channels should be prioritized for relay selection to benefit more viewers.
- Cost-efficient path: The relay path should be selected optimally to reduce the streaming delivery cost in the path.

C. Hybrid Assignment

We further introduce the hybrid assignment solution in details, which determine the relay assignment at different time scales and broadcaster number scales. The hybrid solution determines whether a broadcast streaming should be relayed, and further which relay to use. The centralized assignment takes the whole network information as input and calculates the optimal relay assignment of all broadcasters as output. Due to relatively large computation, the centralized assignment operates in a periodic way. We should strike the tradeoff between system optimality and computation as well as network overhead. Detailed discussion on the period is provided in the experiment section. We formulate the centralized assignment problem as an optimization problem and provide an efficient fast algorithm in Section IV.

When a broadcast channel is established, it should be relayed to an optimal path at once. The centralized assignment is not appropriate in this scenario, as the centralized assignment cannot guarantee fast response. Hence, we design the distributed assignment, which makes a quick decision for a better network condition in subsecond response time when a broadcast channel is established. The relay decision is performed on the broadcasters' devices and relay nodes in a distributed way. Specifically, the broadcaster's device decides which relay to use with the MAB method, which explores the network conditions based on the historical data. Once a broadcast channel is assigned to a relay, the relay nodes then decide which upServer to upload based on the current traffic information. The algorithm is provided in Section V.

D. An Illustrative Example

To introduce the architecture features and demonstrate the key design principles (prioritized broadcast channel and cost-efficient path), we provide an illustrative example in Fig. 13, where we consider a system with two broadcasters $\{B_1, B_2\}$, two relays $\{R_1, R_2\}$, and one upServer $\{U\}$. The bitrates of the broadcasters are $[800, 400]$ kbps. The available capacity of the relay-upServer can be measured, which are $[1000, 800]$ kbps. The link cost of each node pair is denoted in Fig. 13, representing the QoS loss caused by packet loss and transmission delay. Since both the loss rate and delay are additive, we can derive the path cost table in Table II.

We show the optimal relay assignment policy in Fig. 13, where the broadcaster popularities are [1000, 10] and [10, 10], respectively. When the popularity of B_1 is very large, it gets the priority to select the most cost-efficient path $[B_1, R_1, U]$, although B_2 can obtain the larger cost decrease when selecting R_1 as the relay. When the popularity of B_1 and B_2 are the same, B_2 is assigned to R_1, as the path cost is reduced most remarkably.

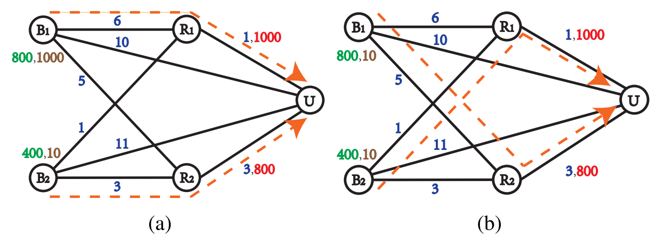


Fig. 13. An illustrative example of relay assignment optimization. The blue values are the link costs, the red values are the available bandwidths, the green values are the required bandwidths, and the brown values are the channel popularities.

TABLE II
RELAY COST MATRIX. (NOTE THAT ϕ MEANS NO RELAY IS USED, AND THE BROADCASTERS UPLOAD TO THE UPSERVER DIRECTLY)

-	ϕ	R_1	R_2
B_1	10	7	8
B_2	11	2	6

IV. CENTRALIZED ASSIGNMENT: FORMULATION AND OPTIMIZATION

In this section, we provide the formulation of the global relay assignment problem. The problem is unique in the following aspects: (1) Optimizing one broadcaster can benefit all the viewers, and the key challenge is estimating the viewer number. (2) Our method can avoid the overload of the upServers as we consider the transcoding and networking capacities. We prove that the optimization problem is NP-hard and design fast algorithms to achieve sub-optimal performance. In order to achieve low latency, low loss rate, and sustainable bandwidth at the same time, which guarantees good network quality, we incorporate the latency and the loss rate as a comprehensive network cost and set the sustainable bandwidth as the constraint. As the centralized assignment is computation-intensive, it works in a periodic way.

A. Basic Network Model

The broadcaster set can be denoted as $\mathcal{B} = \{1, 2, \ldots, B\}$. The edge relays, i.e., devices with network transmission capacity, are used to relay data from the broadcasters to the upServer, and we denote the relay set as $\mathcal{R} = \{1, 2, \ldots, R\}$. The destination of the broadcaster streams is the upServers denoted as $\mathcal{U} = \{1, 2, \ldots, U\}$. We assume that the upServer has the limit of computation, e.g., transcoding, and we denote the limit as Com, which is measured in bps. As we allow one-hop relay, the broadcaster will be assigned to one relay or no relay, depending on the relay assignment. The relay assignment option of each broadcaster b is denoted as $r(b) \in \mathcal{R} \cup \{0\}$, where $r(b) = 0$ means that the broadcaster is assigned to no relays, and delivers the stream to the upServer directly.

Thus, we can denote the relay option set as $\mathcal{R}^* = \mathcal{R} \cup \{0\} = \{0, 1, 2, \ldots, R\}$. Whether or not the broadcaster is assigned to a relay, the destination of the broadcaster stream is one specific upServer of the CDN. Specifically, the upServer assignment of each broadcaster b is denoted as $u(b) \in \mathcal{U}$. The centralized assignment aims to determine the relay and upServer assignment $(r(b), u(b)); \forall b \in \mathcal{B}$ to optimize the global system performance.

B. Network Cost Model

The relay assignment problem relies on the network quality of the broadcaster-to-relay link, the broadcaster-to-upServer link, and the relay-to-upServer link. We denote the available bandwidth between relay r and upServer u as $c(r, u)$. Hence, $C^{R-U} = \{c(r, u)\}; \forall r \in \mathcal{R}, u \in \mathcal{U}$ is a matrix representing the available bandwidths of all possible relay-to-upServer pairs. Note that we cannot derive the available bandwidth related to the broadcaster, as the broadcaster is not under control and the bandwidth is highly dynamic. However, we can estimate the loss rate and delay between any node pair of the uploading path, as mentioned in [11]. For a node pair (i, j) in the uploading path, the loss rate is denoted as $l(i, j)$, and the delay is denoted as $d(i, j)$. Based on above definitions, we define the video QoS of the node pair as a weighted sum of loss rate and delay measured [14]–[16]. Formally, the link cost $s_{i,j}$ can be formulated as follows:

$$s_{ij} = \alpha d(i, j) + (1-\alpha) l(i, j), \quad 0 < \alpha < 1, \ \forall i, j \in \mathcal{B} \cup \mathcal{R} \cup \mathcal{U}, \tag{2}$$

where α is a parameter balancing the loss rate and the delay. The uploading path cost is defined as $S(b, r(b), u(b))$, reflecting the loss rate and transmission latency of the link:

$$S(b, r(b), u(b)) = \begin{cases} s(b, u(b)) & r(b) = 0 \\ s(b, r(b)) + s(r(b), u(b)) & r(b) \in \mathcal{R}^*/\{0\} \end{cases} \tag{3}$$

C. Broadcaster Model

The broadcasters are heterogeneous in bitrate, as each broadcaster has unique device setting and video content.[1] We denote the bitrate of broadcaster b as $t(b)$. Our objective is to minimize the comprehensive network cost of viewers by improving the broadcaster's network. The unique characteristic of optimizing the broadcaster's network is that each broadcast channel bears different popularity, thus has different impacts on viewers. Based on the analysis of the broadcaster patterns, we notice that the viewer number varies among different broadcasters, which inspires us to illustrate the popularity of broadcasters. Specifically, we define the current viewer number as $P_c(b)$, which is a measure of the channel popularity. However, the centralized assignment works in a periodic way, thus requires future information. Hence, a more accurate popularity method is required. In this paper, we develop a technique to estimate the popularity of a channel by focusing on the weighted viewer number of current viewer number and broadcaster average viewer number. We use the exponential moving average method to predict the viewer number as follows:

$$P(b) = (1 - \beta) P_a(b) + \beta P_c(b), \quad 0 < \beta < 1, \tag{4}$$

where $P(b)$ is the popularity of the broadcaster b, $P_a(b)$ is the average concurrent viewer number of broadcaster b, β is the smoothing factor. Larger values β reduces the smoothing level, and when $\beta = 1$ the popularity is the current viewer number. Thus, the overall viewers' cost of channel b is $P(b) \cdot S(b, r(b), u(b))$.

D. Problem Formulation

Our aim is to minimize the overall cost by assigning the relay decisions in the first mile network. We introduce $x_b(r, u) \in \{0, 1\}, \forall r \in \mathcal{R}^*, \forall u \in \mathcal{U}$ to represent the joint relay and upServer assignment, $\vec{x}_b$ follow the constraint:

$$|\vec{x}_b| = 1, \quad \forall b \in \mathcal{B}, \tag{5}$$

indicating that there is only one path from the broadcaster to the upServer.

In order to guarantee sustainable network bandwidth, we have the following link capacity constraints:

$$\sum_{b \in \mathcal{B}} t(b) \cdot x_b(r, u) \le c(r, u), \quad \forall r \in \mathcal{R}, \ u \in \mathcal{U}, \tag{6}$$

The upServer is computation intense as the video transcoding and forwarding is performed in it. As [18] illustrated that the more popular videos should be transcoded into more replications, we define $H(P(b))$ as a concave increasing function of the computing resource spent when transcoding the streaming of broadcaster b with popularity $P(b)$. Hence, the upServer computing constraint can be denoted as:

$$\sum_{b \in \mathcal{B}} t(b) H(P(b)) \cdot \sum_{r \in \mathcal{R}^*} x_b(r, u) \le Com, \quad \forall u \in \mathcal{U}, \tag{7}$$

Then $S(b, r(b), u(b))$ can be reformulated as $S^*(b, \vec{x}_b)$, which is shown as follows:

$$S^*(b, \vec{x}_b) = \sum_{r \in \mathcal{R}^*} \sum_{u \in \mathcal{U}} S(b, r, u) \cdot x_b(r, u) \tag{8}$$

The global relay assignment problem can be formulated as:

$$\begin{aligned} \min \ & \sum_{b \in \mathcal{B}} \sum_{r \in \mathcal{R}^*} \sum_{u \in \mathcal{U}} P(b) \cdot S(b, r, u) \cdot x_b(r, u) \\ & \text{subject to (5), (6), (7)} \\ & \qquad x_b(r, u) \in \{0, 1\}, \quad \forall b \in \mathcal{B}, \ r \in \mathcal{R}^*, \ u \in \mathcal{U}, \end{aligned} \tag{9}$$

We can achieve the optimal relay assignment via solving the above problem, which is denoted as the optimal assignment problem (OAP).

Theorem 1: The optimal assignment problem (OAP) is NP-hard.

Proof: We prove the NP-hardness of OAP by reduction to the knapsack problem. Suppose there are only one relay node and one upload server in this broadcast system, i.e. $\mathcal{R}^* = \{0, 1\}, \mathcal{U} = \{1\}$. We assume the computation capacity of the server is unlimited, i.e., $Com = +\infty$, thus the constraints in

[1] [17] detected that the bitrates in a CLS platform are highly heterogeneous.

TABLE III
NOTATIONS OF IMPORTANT VARIABLES

Notation	Meaning
$\mathcal{B}$	The broadcaster set
$\mathcal{R}$	The relay set
$\mathcal{U}$	The upServer set
Com	The computation limit of one upServer
$r(b)$	The relay assignment option of broadcaster b
$u(b)$	The upServer assignment option of broadcaster b
$c(r,u)$	The available bandwidth between relay r and upServer u
$l(i,j),\ d(i,j)$	The loss rate/delay between node pair (i,j)
$S(b,r,u)$	The uploading path cost
$t(b)$	The bitrate of broadcaster b
$P_c(b)$	The current viewer number of broadcaster b
$P_a(b)$	The average concurrent viewer number of broadcaster b
$P(b)$	The popularity of broadcaster b
$x_b(r,u) \in \{0,1\}$	The path assignment indicator

equation 7 always satisfies. Furthermore, we assume that all broadcasters taking the relay path bear the same link quality and we normalize it to 1, i.e., $S(b,1,1)=1, \forall b \in \mathcal{B}$ and $S(b,0,1)=0, \forall b \in \mathcal{B}, \forall u \in \mathcal{U}$. Moreover, the constraint of capacity on path $\{1,1\}$ is denoted as $l(1,1)=W$. The problem becomes "How to maximize the viewer number transmitted via relay network under the constraint of the path capacity, given that each broadcaster bears a bitrate and a viewer number." Specifically, the original problem is reduced to the following problem:

$$\begin{aligned} \max \quad & \sum_{b\in\mathcal{B}} P(b)x_b \\ & \sum_{b\in\mathcal{B}} t(b)x_b \le W \\ & x_b \in \{0,1\}, \quad \forall b \in \mathcal{B}, \end{aligned} \tag{10}$$

We notice that the above problem is a classical 0-1 knapsack problem. As the 0-1 knapsack problem is known to be NP-hard, we prove the NP-hardness of the OAP.

E. Algorithms

Since OAP is proved to be NP-hard, we cannot obtain the optimal solution in polynomial time. In this section, we provide Greedy Rounding Algorithm (GRA) for the OAP with rounding technique, and prove that the method is theoretically bounded. GRA method runs relatively fast in small and medium network scale, while in the large-scale network, we need faster implementation. In order to accelerate the calculation process, we further develop a fast implementation (FGRA) of GRA, which has computing complexity in polynomial time and processes fast in the large-scale network.

1) Greedy Rounding Algorithm: Intuitively, we prioritize the optimization of popular channels to benefit more viewers. However, the direct greedy strategy may not provide satisfying results as it ignores the network quality. A better algorithm should be designed to consider both the channel popularity and the network resource. Motivated by the rounding techniques [19], we design a greedy rounding algorithm (GRA).

$$x_b(r,u) \ge 0, \quad \forall b \in \mathcal{B},\ r \in \mathcal{R}^*,\ u \in \mathcal{U} \tag{11}$$

First we relax the binary variant constraints as shown in equation (11). This relaxation changes the original problem into a linear programming problem. This relaxed linear programming can be effectively solved using classical methods like simplex method [20]. In this way, we can derive the solution of the linear programming denoted as $x_b^*(r,u)$. This solution cannot be applied directly, as we expect binary variants as the solution. We further need to obtain a feasible binary variables solution.

Here we extract the system cost brought up by each broadcaster as follows:

$$W(x_b^*(r,u)) = P(b)\cdot S(b,r,u)x_b^*(r,u), \tag{12}$$

which can be regarded as the "weight" of the broadcaster. This weight reflects how much cost the variable induces. Intuitively, the variables inducing larger cost should be set at a smaller value. Hence, we round the broadcasters in weight descending order. Furthermore, in order to satisfy the constraint of equation (5), we can only round $x_b(r,u)$ with the highest weight. Specifically, we show the rounding policy as follows:

$$x_b(r,u) = \begin{cases} 0 & \text{if } (r,u) \neq \arg_{(r,u)} \max W(x_b^*(r,u)) \\ 1 & \text{if } (r,u) = \arg_{(r,u)} \max W(x_b^*(r,u)) \end{cases} \tag{13}$$

After performing the rounding process, the link capacity and computation constraints may still be violated. Thus we should consider the feasibility in the rounding process, and round the broadcaster which is feasible and has the largest weight. We show the greedy rounding algorithm in Algorithm 1.

We can prove that GRA is $\mathcal{O}(|\mathcal{R}|\cdot|\mathcal{U}|)$-optimality in cost upper bound.

2) Fast Implementation for Greedy Rounding Algorithm (FGRA): We notice that in GRA, we need to solve a linear programming in the relaxation part. The most widely accepted method is the simplex method, which may induce exponential complexity [20] when the network scale gets large. As a result, we develop FGRA, which uses a heuristic to approximate GRA method. We look into $x_b^*(r,u)$ in the relaxed linear programming problem, and find that the path with lower cost corresponds to a larger $x_b^*(r,u)$. Heuristically, we want to find an approximation for $x_b^*(r,u)$. We define γ as the lowest achieved cost without relay:

$$\gamma = \min_u (S(b, R+1, u)), \quad \forall u \in \mathcal{U} \tag{14}$$

Then we use heuristic to define $x_b^*(r,u)$:

$$x_b^*(r,u) = e^{\gamma - S(b,r,u)}, \quad \forall b \in \mathcal{B},\ r \in \mathcal{R}^*,\ u \in \mathcal{U}. \tag{15}$$

Algorithm 1 Greedy Rounding Algorithm

```
1: procedure GREEDY-ROUNDING($\boldsymbol{P}, \boldsymbol{l}, \boldsymbol{S}, \boldsymbol{t}, Com$)
2:     I. Relaxation
3:     Solve the relaxed LP and get fraction results $x_b^*(r,u)$
4:     II. Rounding
5:     for $b \in \mathcal{B}$ ranked by $\sum_{r,u} P(b)S(b,r,u)x_b^*(r,u)$ do
6:         Rank the $(r,u), r \in \mathcal{R}^*, u \in \mathcal{U}$ by $W(x_b^*(r,u))$
7:         $(r_b, u_b) \leftarrow$ the first $(r,u)$ that satisfies the const.
8:         $x_b(r_b, u_b) \leftarrow 1$
9:         for $(r,u) \neq (r_b,u_b), r \in \mathcal{R}^*, u \in \mathcal{U}$ do
10:            $x_b(r,u) \leftarrow 0$
11:        end for
12:    end for
13: end procedure
```

After obtaining $x_b^*(r,u)$, the rounding process in FGRA method is the same as the GRA method. We have the following theorem for the complexity of the heuristic greedy rounding algorithm (FGRA).

Theorem 2: The computing complexity of the FGRA algorithm is $\mathcal{O}(|\mathcal{B}| \cdot |\mathcal{U}| \cdot |\mathcal{R}| \log(|\mathcal{U}| \cdot |\mathcal{R}|))$.

Proof: We consider the complexity of rounding process - the second part in the algorithm. The time limit appears at the step of ranking the (r,u) pairs for each $b \in \mathcal{B}$. The complexity of this rounding part for one broadcaster is $\mathcal{O}(|\mathcal{U}| \cdot |\mathcal{R}| \log(|\mathcal{U}| \cdot |\mathcal{R}|))$. Thus the complexity of all the broadcasters is $\mathcal{O}(|\mathcal{B}| \cdot |\mathcal{U}| \cdot |\mathcal{R}| \log(|\mathcal{U}| \cdot |\mathcal{R}|))$.

V. Distributed Assignment: MAB-Based Method

As the broadcast channels establish as well as terminate frequently, and the network condition varies with time, a real-time assignment is necessary for quick decision making to deal with the broadcast channel dynamic. Due to the time-consuming computation of global optimization, the centralized assignment cannot guarantee fast response of a newly established channel. In this way, we design a distributed assignment method for broadcasters when a broadcast channel is established. The assignment is made by the broadcast device and relays in a distributed two-phase manner. In phase I, the broadcaster uses MAB-based method for relay assignment based on the historical network performance. In phase II, the relay forwards the streaming to the optimal upServer based on instant relay load. The reason that we use MAB method is that it can characterize the network performance in the time-variant condition.

A. Phase I: Broadcaster Local Assignment

1) Multi-Armed Bandit Problem: The MAB problem [7] models an agent with historical information taking action repeatedly in order to maximize the accumulated reward. It works in an *exploration and exploitation* way, i.e., taking the currently best action as well as acquiring new information. The classic MAB problem considers a bandit with F arms, whose expected rewards are i.i.d. over time. At each time slot, one arm is pulled and the bandit yields a reward. The problem is to decide which arm to pull at each time slot in order to maximize the accumulated expected reward over time. An arm is more reliable if it is pulled more times, while pulling an arm with higher expected rewards introduces higher reward. Ideally, we want to pull the arms with reliable information and high expected rewards. Hence, there is a tradeoff between the exploration of new arms (i.e., pulling an more times to derive accurate estimation of the mean rewards) and the exploitation of known arms (i.e., obtaining higher rewards).

We can see a natural mapping between the MAB problem and the relay selection problem for broadcasters. Like MAB, each time when a broadcast channel is established and uses a relay (i.e., pull a bandit), we can observe the induced network cost (i.e., reward). Our goal is to minimize the overall network cost (i.e., reward maximization). Threr are many algorithms for the MAB problem. Reference [21] shows that no policy can achieve an asymptotic regret smaller than $O(log(t))$. In this way, we select a policy proposed in [22], i.e., the upper confidence bounds (UCB), which is proven to achieve a regret on the order of $O(log(t))$.

2) MAB for Relay Selection: For a particular broadcaster b, our goal is to select a relay with the lowest cost. As the network condition is highly dynamic, the goal for relay selection is to reach a balance between exploring new relays and exploiting the currently optimal relay. The UCB method considers both how close the choice is to optimality and the skewness historical selection. In this way, the algorithm will explore the relay nodes which are seldom used and exploit the current optimal relay selection scheme.

Formally, we define the MAB problem for each broadcaster as follows. Recall that R^* is the set of relay options and $s(b,r)$ is the expected network cost when using r as the relay option. We denote $A \in \mathcal{R}^*$ as the selected relay, $Q(A)$ as the estimated action value when performing A and R as the actual value after acting A. In our scenario, R can be measured as the network cost $s(b,r)$. Furthermore, we denote $N(A)$ as the number that a relay node has been chosen before. The relay selection scheme is based on both the estimated value $Q(r)$ and the exploring factor $\sqrt{\frac{\log t}{N(r)}}$, where t is the number of historical relay selections. We treat the weighted combination of the two factors as the objective function:

$$A = \arg\min_r Q(r) + c\sqrt{\frac{\log t}{N(r)}}, \quad c < 0. \tag{16}$$

Different from the classical MAB problem aiming to maximize the overall reward, we aim to minimize the network cost $s(b,r)$ in our context. $Q(A)$ is updated by adding $\frac{1}{N(A)}(R - Q(A))$ after choosing A as the action. Algorithm 2 shows our approach. Note that when we apply the MAB-UCB method to select the optimal bandwidth, we maximize the expected bandwidth as in the classical problem.

B. Phase II: Relay Local Assignment

In Phase I, a newly launched broadcast channel will be relayed directly to the upServer or the relay. Once a channel is forwarded to a relay, a further decision should be made as to which upServer to assign. This decision is made by the relay node that the streaming is assigned to based on the

Algorithm 2 Multi-Armed Bandit With Upper-Confidence-Bound

```
1:  procedure MAB-UCB($b$, $R^*$, $A$)
2:      I. Initialization
3:      $t \leftarrow 1$
4:      for $r = 1$ to $R^*$ do
5:          $Q(r) \leftarrow 0$
6:          $N(r) \leftarrow 0$
7:      end for
8:      II. Recursion
9:      $A \leftarrow \arg\min_r Q(r) + c\sqrt{\frac{\log t}{N(r)}}$
10:     $R \leftarrow bandit(A)$
11:     $t \leftarrow t + 1$
12:     $N(A) \leftarrow N(A) + 1$
13:     $Q(A) \leftarrow Q(A) + \frac{1}{N(A)}(R - Q(A))$
14: end procedure
```

link capacity between the relay node and the upServers and work load on the upServers. The relay can obtain the optimal solution of upServer selection by computing the potential cost when selecting different upServers. The relay can filter all available links whose capacity is larger than the streaming bitrate and choose the link with the best network performance.

VI. Experiment Result

A. Experiment Setup

In order to validate the optimality and the processing efficiency of the proposed algorithms, we set up an evaluation environment based on the real-trace. We select 1000 broadcasters from the Inke.tv data trace, i.e., $B = 1000$, whose bitrates range from 1Mbps to 3Mbps, and set $U = 4$ as the upServer number. We also set the computing limitation of each upServer is 2,000 Mbps, the default relay number as 100, i.e., $R = 100$, and the default α as 0.4. We collected the dataset of smart routers in Beijing City, and treat the routers as the relay servers. The dataset contains the router ID, the location, and the network performance. After deriving the performance of the algorithms in the simulation environment, we run the real-world experiment on the whole Inke.tv dataset, aiming to evaluate the potential performance improvement on the real system.

We implement six methods of centralized assignment for comparison, i.e., RANDOM, VDN-C, TOP-N, GRA, FGRA, OPT. RANDOM is assigning channels to upServers randomly without the aid of relay network. VDN-C [23] is a centralized assignment method to allocate resource optimally in the live streaming system without relay network, and serves as the baseline. TOP-N uses the relay network and assigns the broadcasters sequentially by the popularity of channels. TOP-N also serves as the baseline, which is an intuitive assignment method for the relay assignment. OPT is the theoretically optimal solution of the relay assignment problem. We implement three distributed methods, i.e., Static, Greedy, UCB. The "Static" method chooses the same relay all the time. The "Greedy" method selects the currently optimal relay based on historical data.

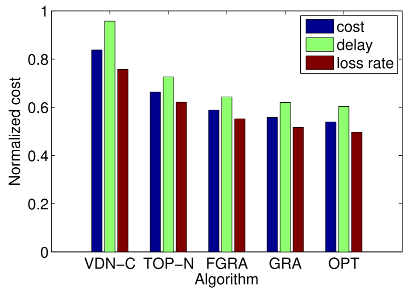


Fig. 14. Normalized Cost of different algorithms.

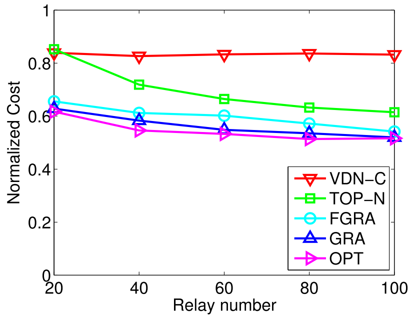


Fig. 15. Normalized Cost versus relay number.

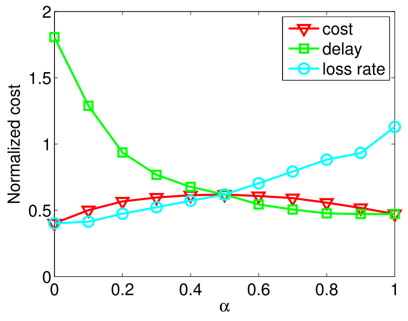


Fig. 16. Cost, delay, and loss rate with α.

B. Simulation Results

In order to simplify the analysis, we normalize the cost of viewers to the range from 0 to 1 by dividing the network cost of the RANDOM algorithm. We now present the viewers' normalized costs under five methods in Fig. 14. We notice that with the implement of the relay network, the viewer cost can be reduced by 25% $\sim$ 43%, as the VDN-C method has the highest cost, delay, and loss rate. Recall that the cost is the QoE loss of all viewers. The OPT method has the lowest cost and is regarded as the optimal baseline. TOP-N method induces the highest cost in relay-based methods. The costs of GRA and FGRA are very close, and the cost of FGRA is slightly higher that GRA. This suggests that the heuristics do not cause much accuracy loss. Compared with OPT, GRA incurs only 3% more cost, and FGRA incurs 8% more cost. Meanwhile, TOP-N incurs 25% more cost than the OPT method.

We show the viewer cost versus relay number in Fig. 15, where we randomly remove some relays and calculate the

TABLE IV
TOTAL COST VERSUS PERIOD OF CENTRALIZED ASSIGNMENT

Period	5min	10min	15min	20min	30min	1hour
Cost ($\times 10^7$)	3.19	3.89	4.07	4.68	5.31	7.26

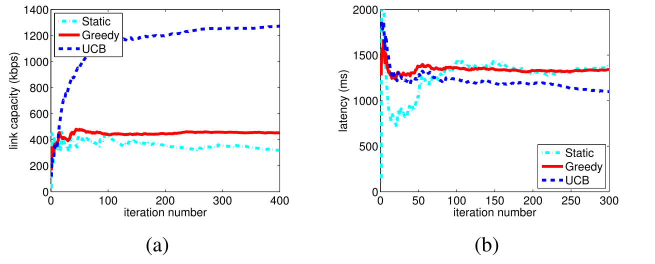


Fig. 17. Iteration processes of the MAB algorithm for link capacity and latency selection. (a) Iteration process in distributed assignment to maximize link capacity. (b) Iteration process in distributed assignment to minimize latency.

total cost of viewers. As the relay is not used in the VDN-C algorithm, the network cost of VDN-C remains the same. Other relay-based algorithms can achieve lower cost when more relay nodes are employed because broadcasters have a larger probability to find a cost-efficient relay path with more relays. Both the GRA and FGRA algorithms can reach 95% of the performance of the OPT method.

Furthermore, we discuss the cost, delay, and loss rate versus α in Fig. 16, and the employed algorithm is FGRA. In reality, the value of α can be arbitrarily set to balance the weights of the multiple objectives: low latency and low loss rate. For example, when a low delay is expected, we can set a larger value for α to improve the weight of delay. We notice that with the increase of α, the transmission delay decreases and the loss rate increases.

We show the total cost as a function of the refresh period of the centralized assignment. Table. IV show the total cost with the deployment of only the centralized assignment under different refresh period. For the fair comparison, the total cost is calculated in a day based on the same set of broadcasts and views. We observe that the finer granularity results in higher cost reduction. Note that in reality, the period should be set considering the trade-off between the cost reduction and the computation overhead.

Next, we investigate the performance of the MAB algorithm in the distributed relay selection. We show the performance under different iteration times of different distributed assignment algorithms in Fig. 17(a) and Fig. 17(b), respectively. We have two key observations: First, we find that the MAB-based method outperforms baselines significantly, i.e., MAB method can improve the average link capacity by 140% compared to the greedy algorithm, and the average link capacity can reach 1270 *kbps*. MAB can also lower the transmission latency by 27% compared to baselines. Second, the convergence of the MAB algorithm is relatively fast. The MAB algorithm converges in around 100 iterations both in finding the maximal bandwidth and the lowest latency.

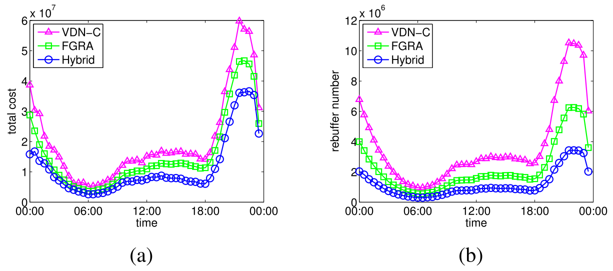


Fig. 18. Experimental results on a typical day of the Inke dataset. (a) Total cost in a day. (b) Rebuffer number in a day.

Also, MAB requires smaller iteration numbers to outperform the baselines: MAB outperforms baselines in 20 iterations for high bandwidth and in 75 iterations for low latency.

C. Real-Trace Experiment

We investigate the total cost of the Inke.tv platform in a typical day with different methods in Fig. 18(a). For fairness comparison, we set the refresh period as 30 minutes for all methods. As the VDN-C method do not utilize the relay network, the VDN-C method induces the highest cost of all time and serves as the benchmark. Deploying only the Centralized assignment method (we choose FGRA for practical consideration) induces lower cost than VDN-C. The hybrid assignment method utilizing centralized method (FGRA) and distributed method (UCB) achieves the best performance. Comparing with the pure FGRA, the gap of cost enlarges during peak demand and reaches 20% cost reduction. This is because in peak demands, the dynamic of broadcast channels increase, thus more new-launched broadcast channels will be assigned by distributed assignment. The hybrid method outperforms the VDN-C method by 40% in cost reduction. We further investigate the viewer rebuffer behavior based on the measurement result provided in Section II, from which we can infer the rebuffer number in a broadcast channel by broadcaster uploading latency. Specifically, we can compute the uploading latency for each channel, and estimate the viewer rebuffer number based on equation (1). Then we sum up the estimated rebuffer number of all broadcast channels to derive the total rebuffer number in the system. Note that the rebuffer number estimation for each channel may not be accurate, while according to the law of large numbers, the rebuffer number estimation of the whole system is convincing. We provide the rebuffer number on the same day with different methods in Fig. 18(b). We find the viewer rebuffer number will decrease dramatically when the relay-based methods are utilized. We find that a great portion of rebuffers occur in the peak hours, and the FGRA method can decrease the rebuffer number by 40%. In addition, we find the Hybrid method (FGRA+distributed assignment) can further decrease the rebuffer number by 10%, as the distributed assignment can select a better path when the channel is established, thus benefiting the earliest viewers to the channel.

TABLE V
PROCESSING TIME VERSUS RELAY NUMBER

Relay Number	Processing Time (sec)			
	TOP-N	FGRA	GRA	OPT
200	0.005	0.012	6.752	185.542
400	0.009	0.020	13.552	359.121
600	0.011	0.033	23.312	NA
800	0.016	0.047	34.316	NA
1000	0.020	0.058	44.963	NA

TABLE VI
PROCESSING TIME VERSUS BROADCASTER NUMBER

Broadcaster Number	Processing Time (sec)			
	TOP-N	FGRA	GRA	OPT
10^1	0.002	0.003	0.879	1.804
10^2	0.012	0.025	9.333	19.003
10^3	0.109	0.361	98.562	208.653
10^4	1.254	3.562	NA	NA
10^5	14.432	39.671	NA	NA

D. Processing Time Analysis

Above results indicate the relay network is notable for reducing the total cost and viewer rebuffer time. As a real-world online implementation, another important metric is the time consumption in scheduling. On one hand, the network condition shifts quickly, thus scheduling should be refreshed in a few minutes. On the other hand, the number of broadcasters and relays may be very large in reality, challenging the processing speed. We realize the centralized assignment algorithm in C++ for fast implementation. Then we present the processing time of different methods with the number of relay and broadcaster in Table. V and Table. VI, respectively. We find that the processing time of FGRA is 100x$\sim$1000x faster than GRA. It takes about 39 seconds to finish the assignment process when the broadcaster number reaches $100,000$. The running time measurement result was conducted on a desktop with Intel Core i7-6700 CPU @ 2.6 GHz x 4.

VII. RELATED WORK

A. Live Streaming System Optimization

With the growing popularity among users, CLS has also attracted attention from the academia. Reference [24] and [25] conducted measurement study on live streaming systems like Meerkat and Periscope, and found that nowadays the typical live streaming service employs CDN to deliver contents. Many previous works focus on the optimization of the live streaming system. Mukerjee *et al.* [23] designed a hybrid controller to improve the performance of the CDN, and further used a centralized algorithm for live video optimization. Wu *et al.* [26] proposed a collaborative transcoding strategy for live streaming. Chen *et al.* [27] presented a generic framework that facilitates a cost-effective cloud service for crowdsourced live streaming. Wang *et al.* [28] proposed CALMS, which adaptively leases and adjusts cloud server resources. He *et al.* [17] introduced a framework that utilized cloud computing services to enhance the viewer satisfaction and allocate the geo-distributed computing resources. Reference [29] forecasted highlight in video game streaming to assist in bitrate adaptation. Reference [30] aimed to use scalable video coding in mobile devices for live streaming uploading. Although the authors try to solve the uploading problem, they focused on the end device, and did not try to improve the network quality from the device to the server. Gao *et al.* [31], [32] investigated the video transcoding in the cloud.

Most above mentioned works on live streaming focus on the optimization of the cloud network [17], the intra-CDN link [23], and the uploading device end [30]. None of the works focus on the optimization of the first-mile upload network. While, measurement study in [8] showed that the dynamic uploading capacity of broadcasters is a critical challenge, which noticeably affects the smoothness for viewers. Reference [33] conducted data measurement on YouTube Live and Twitch, and found numerous unique uploaders can guarantee 24/7 service. Reference [34] implemented a broadcaster uploading system based on the HTTP protocal, from which we observe that the uploading path is essential for better video quality and lower latency. This inspires us to design novel architecture to improve the first-mile network.

B. Relay Network for Better Quality

Relay network has been applied in many scenarios, such as virtual private networks (VPNs) and multicast [35], [36]. Savage *et al.* [10] found that in 30-80% of the cases, there is an alternate path with significantly superior quality. During last few years, relay network has been used in novel network applications like Internet Telephony and live streaming. Jiang *et al.* [11] presented overlay network to improve the quality of internet telephony call. They used data analysis and machine learning methods to probe the network condition and schedule the relay decision. Zhang *et al.* [37] designed a data-driven overlay network to improve live streaming quality. They use the relay network in the P2P and server-client scenarios. However with the advent of CLS, employing the relay network to optimize the first-mile upload network in CLS system is more important. Reference [38] designed an optimal relay selection algorithm for the wireless relay network.

VIII. CONCLUSION

In this paper, we perform measurement on a large-scale CLS dataset, and the results show that the first mile network causes the viewer QoE degradation. Motivated by this, we design a relay-assistant network for content harvesting in the first mile network for crowdsourced live streaming. We use a hybrid solution, i.e., joint centralized and distributed assignment, to perform the relay assignment. We model the centralized relay assignment as an optimization problem and developed an optimal algorithm and a fast approximation algorithm. We utilize the MAB-based method to perform the distributed assignment locally. The performance of the proposed solution is evaluated through extensive experiments.

REFERENCES

[1] "Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021," Cisco, San Jose, CA, USA, White Paper, 2016.

[2] (2018). *Inke.tv*. [Online]. Available: http://www.inke.tv/

[3] J. He, D. Wu, Y. Zeng, X. Hei, and Y. Wen, "Toward optimal deployment of cloud-assisted video distribution services," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 10, pp. 1717–1728, Oct. 2013.

[4] H. Pang, Z. Wang, C. Yan, Q. Ding, and L. Sun, "First mile in crowdsourced live streaming: A content harvest network approach," in *Proc. Thematic Workshops ACM Multimedia*, 2017, pp. 101–109.

[5] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[6] L. Chen, Y. Zhou, M. Jing, and R. T. B. Ma, "Thunder crystal: A novel crowdsourcing-based content distribution platform," in *Proc. 25th ACM Workshop Netw. Oper. Syst. Support Digit. Audio Video*, 2015, pp. 43–48.

[7] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and non-stochastic multi-armed bandit problems," *Found. Trends Mach. Learn.*, vol. 5, no. 1, pp. 1–122, 2012.

[8] C. Zhang and J. Liu, "On crowdsourced interactive live streaming: A Twitch.tv-based measurement study," in *Proc. 25th ACM Workshop Netw. Oper. Syst. Support Digit. Audio Video*, 2015, pp. 55–60.

[9] M. Ma, Z. Wang, K. Su, and L. Sun, "Understanding content placement strategies in smartrouter-based peer video CDN," in *Proc. 26th Int. Workshop Netw. Oper. Syst. Support Digit. Audio Video*, 2016, p. 7.

[10] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson, "The end-to-end effects of Internet path selection," *SIGCOMM Comput. Commun. Rev.*, vol. 29, no. 4, pp. 289–299, Aug. 1999.

[11] J. Jiang *et al.*, "Via: Improving Internet telephony call quality using predictive relay selection," in *Proc. ACM SIGCOMM Conf.*, 2016, pp. 286–299.

[12] M. Li, Y.-L. Wu, and C.-R. Chang, "Available bandwidth estimation for the network paths with multiple tight links and bursty traffic," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 353–367, 2013.

[13] A. Ganjam *et al.*, "C3: Internet-scale control plane for video quality optimization," in *Proc. NSDI*, vol. 15, 2015, pp. 131–144.

[14] R. K. P. Mok, E. W. W. Chan, and R. K. C. Chang, "Measuring the quality of experience of HTTP video streaming," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2011, pp. 485–492.

[15] H. E. Egilmez, S. Civanlar, and A. M. Tekalp, "An optimization framework for QoS-enabled adaptive video streaming over OpenFlow networks," *IEEE Trans. Multimedia*, vol. 15, no. 3, pp. 710–715, Apr. 2013.

[16] Q. He, C. Zhang, and J. Liu, "Utilizing massive viewers for video transcoding in crowdsourced live streaming," in *Proc. IEEE 9th Int. Conf. Cloud Comput. (CLOUD)*, Jun./Jul. 2016, pp. 116–123.

[17] Q. He, J. Liu, C. Wang, and B. Li, "Coping with heterogeneous video contributors and viewers in crowdsourced live streaming: A cloud-based approach," *IEEE Trans. Multimedia*, vol. 18, no. 5, pp. 916–928, May 2016.

[18] Z. Wang, L. Sun, C. Wu, W. Zhu, and S. Yang, "Joint online transcoding and geo-distributed delivery for dynamic adaptive streaming," in *Proc. IEEE INFOCOM*, Apr./May 2014, pp. 91–99.

[19] P. Raghavan and C. D. Tompson, "Randomized rounding: A technique for provably good algorithms and algorithmic proofs," *Combinatorica*, vol. 7, no. 4, pp. 365–374, 1987.

[20] V. Klee and G. J. Minty, "How good is the simplex algorithm," Dept. Math., Univ. Washington, Seattle, WA, USA, Tech. Rep. AD0706119, 1970.

[21] T. L. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Adv. Appl. Math.*, vol. 6, no. 1, pp. 4–22, Mar. 1985.

[22] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2, pp. 235–256, 2002.

[23] M. K. Mukerjee, D. Naylor, J. Jiang, D. Han, S. Seshan, and H. Zhang, "Practical, real-time centralized control for CDN-based live video delivery," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 311–324, 2015.

[24] B. Wang, X. Zhang, G. Wang, H. Zheng, and B. Y. Zhao, "Anatomy of a personalized livestreaming system," in *Proc. Internet Meas. Conf.*, 2016, pp. 485–498.

[25] M. Siekkinen, E. Masala, and T. Kämäräinen, "A first look at quality of mobile live streaming experience: The case of periscope," in *Proc. Internet Meas. Conf.*, 2016, pp. 477–483.

[26] J.-C. Wu, P. Huang, J. Yao, and H. H. Chen, "A collaborative transcoding strategy for live broadcasting over peer-to-peer IPTV networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 2, pp. 220–224, Feb. 2011.

[27] F. Chen, C. Zhang, F. Wang, J. Liu, X. Wang, and Y. Liu, "Cloud-assisted live streaming for crowdsourced multimedia content," *IEEE Trans. Multimedia*, vol. 17, no. 9, pp. 1471–1483, Sep. 2015.

[28] F. Wang, J. Liu, M. Chen, and H. Wang, "Migration towards cloud-assisted live media streaming," *IEEE/ACM Trans. Netw.*, vol. 24, no. 1, pp. 272–282, Feb. 2016.

[29] W.-T. Chu and Y.-C. Chou, "On broadcasted game video analysis: Event detection, highlight detection, and highlight forecast," *Multimedia Tools Appl.*, vol. 76, no. 7, pp. 9735–9758, 2017.

[30] M. Siekkinen, E. Masala, and J. K. Nurminen, "Optimized upload strategies for live scalable video transmission from mobile devices," *IEEE Trans. Mobile Comput.*, vol. 16, no. 4, pp. 1059–1072, Apr. 2017.

[31] G. Gao, H. Hu, Y. Wen, and C. Westphal, "Resource provisioning and profit maximization for transcoding in clouds: A two-timescale approach," *IEEE Trans. Multimedia*, vol. 19, no. 4, pp. 836–848, Apr. 2017.

[32] G. Gao, W. Zhang, Y. Wen, Z. Wang, and W. Zhu, "Towards cost-efficient video transcoding in media cloud: Insights learned from user viewing patterns," *IEEE Trans. Multimedia*, vol. 17, no. 8, pp. 1286–1296, Aug. 2015.

[33] K. Pires and G. Simon, "YouTube live and Twitch: A tour of user-generated live streaming systems," in *Proc. 6th ACM Multimedia Syst. Conf.*, 2015, pp. 225–230.

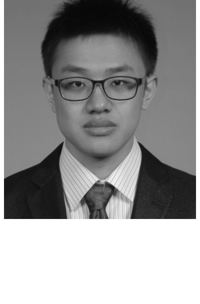
[34] B. Seo, W. Cui, and R. Zimmermann, "An experimental study of video uploading from mobile devices with HTTP streaming," in *Proc. 3rd Multimedia Syst. Conf.*, 2012, pp. 215–225.

[35] Y. Xu, D. Hu, and S. Mao, "Relay-assisted multiuser video streaming in cognitive radio networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 10, pp. 1758–1770, Oct. 2014.

[36] J. Tang, W. P. Tay, and Y. Wen, "Dynamic request redirection and elastic service scaling in cloud-centric media networks," *IEEE Trans. Multimedia*, vol. 16, no. 5, pp. 1434–1445, Aug. 2014.

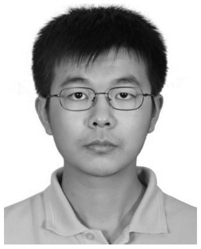
[37] X. Zhang, J. Liu, B. Li, and Y.-S. P. Yum, "CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming," in *Proc. IEEE 24th Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM)*, vol. 3, Mar. 2005, pp. 2102–2111.

[38] D. Yang, X. Fang, and G. Xue, "OPRA: Optimal relay assignment for capacity maximization in cooperative networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2011, pp. 1–6.



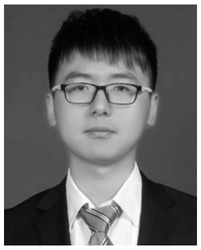
Haitian Pang (S'16) received the B.E. degree from the Department of Automation, Tsinghua University, Beijing, China, in 2014, where he is currently pursuing the Ph.D. degree in computer science. His research areas include network game modeling, cellular-WiFi networking, video streaming system design, and mobile networking optimizations.

Zhi Wang (S'10–M'14) received the B.E. and Ph.D. degrees in computer science from Tsinghua University, Beijing, China, in 2008 and 2014, respectively. He is currently an Assistant Professor with the Graduate School at Shenzhen, Tsinghua University. His research areas include online social networks, mobile cloud computing, and large-scale multimedia systems. He is a recipient of the ACM Multimedia Best Paper Award (2012), the China Computer Federation (CCF) Outstanding Doctoral Dissertation Award (2014), and the MMM Best Student Paper Award (2015).



Chen Yan received the B.S. degree from the Department of Computer Science and Technology, Tsinghua University, in 2015, where he is currently pursuing the master's degree. His research interests include data mining, and computer vision and their applications in urban computing.

Qinghua Ding is currently pursuing the bachelor's degree with the Multimedia and Networks Laboratory, Tsinghua University. He is a Student Research Fellow (SRT student) Multimedia and Networks Laboratory, Tsinghua University. His research interests include game theory and network economics, optimization of computer networks, and artificial intelligence. He has published some articles on the ACM MM, the IEEE ICC, and the IEEE Globecom.



Kun Yi received the B.S. degree from the Department of Computer Science and Technology, Tsinghua University, in 2017, where he is currently pursuing the master's degree.

Jiangchuan Liu (S'01–M'03–SM'08–F'17) was an Assistant Professor with The Chinese University of Hong Kong, and a Research Fellow at Microsoft Research Asia. He is an EMC-Endowed Visiting Chair Professor with Tsinghua University, Beijing, China, and also an Adjunct Professor of Tsinghua-Berkeley Shenzhen Institute. He is currently a University Professor with the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada. He is an NSERC E.W.R. Steacie Memorial Fellow.

He received the B.Eng. degree *(cum laude)* from Tsinghua University, Beijing, China, in 1999, and the Ph.D. degree from The Hong Kong University of Science and Technology in 2003, both in computer science. He is a co-recipient of the inaugural Test of Time Paper Award of the IEEE INFOCOM (2015), the ACM SIGMM TOMCCAP Nicolas D. Georganas Best Paper Award (2013), and the ACM Multimedia Best Paper Award (2012).

His research interests include multimedia systems and networks, cloud computing, social networking, online gaming, big data computing, RFID, and Internet of things. He is a Steering Committee Member of the IEEE TRANSACTIONS ON MOBILE COMPUTING and the Steering Committee Chair of the IEEE/ACM IWQoS (2015–2017). He has served on the editorial boards of the IEEE/ACM TRANSACTIONS ON NETWORKING, the IEEE TRANSACTIONS ON BIG DATA, the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, and the IEEE INTERNET OF THINGS Journal.

Lifeng Sun (M'05) received the B.S. and Ph.D. degrees in system engineering from the National University of Defense Technology, Changsha, China, in 1995 and 2000, respectively. He was a Post-Doctoral Fellow from 2001 to 2003, an Assistant Professor from 2003 to 2007, and an Associate Professor from 2007 to 2013 with the Department of Computer Science and Technology, Tsinghua University, where he is currently a Professor. His research interests lie in the areas of online social network, video streaming, interactive multi-view video, and distributed video coding. He is a member of the ACM.