Content to Cash: Understanding and Improving Crowdsourced Live Video Broadcasting Services with Monetary Donations

Yuhan Hu[a], Ye Tian[a,*], Wen Yang[a], Xiaodong Wang[a], Xinming Zhang[a]

[a] *Anhui Key Laboratory on High-Performance Computing*
School of Computer Science and Technology, University of Science and Technology of China, Hefei, 230026, Anhui, China

Abstract

Crowdsourced live video broadcasting (livecast) services such as Twitch and Douyu have become increasingly popular in recent years. In such a service, how to allocate limited service capacities, including video transcoding and delivery capacities among numerous channels, is a critical problem. Previous studies allocate capacities based on popularity. In this paper, we analyze Douyu, a leading crowdsourced livecast website in China, with a measurement approach. We find that Douyu is deployed upon a video delivery network (VDN), and it prioritizes popular channels when allocating service capacities; we also find that viewers' willingness to donate monetary gifts in a channel is closely related to their streaming experiences, which are decided by service capacities allocated in the channel. On the other hand, a livecast channel's profitability is only moderately correlated to its popularity. In other words, there exists a mismatch between the popularity-based service strategies and Douyu's business model. Motivated by our analysis, we propose that channels' profitability as well as popularity should be considered in capacity allocating. We present proactive and reactive algorithms, which balance viewers' satisfaction with system's monetary profit, for allocating transcoding capacity among livecast channels. We also propose a practical strategy for VDN edge nodes to select channels to replicate, by taking channels' popularity, profitability, and bandwidth consumptions into consideration. Experiments driven by real-world measurement data show that our proposed solutions can effectively improve the overall benefits for a crowdsourced livecast system and individual VDN edge nodes, and avoid adjusting channels' transcoding schemes too often during livecast sessions.

Keywords: Crowdsourced live video broadcasting; monetary virtual gift donation; service capacity allocation

1. Introduction

Internet crowdsourced live video broadcasting (*livecast* for short) services, which allow anyone to broadcast live videos from anywhere on the Internet, have attracted millions of audience and formed a big entertainment industry in recent years. For example, `Twitch.tv`, the leading crowdsourced livecast website owned by Amazon, has become the fourth largest source of the Internet peak traffic in the US [1]; and `Douyu.com`, the most prominent Chinese crowdsourced livecast website, had attracted over two billion RMB of investment by 2014, and accumulated over 200 million registered users by the year 2017.

*Corresponding author
Email address: `yetian@ustc.edu.cn` (Ye Tian)

Comparing with conventional IPTV systems (e.g., [2] and [3]), a crowdsourced livecast system has several unique characteristics. First, the system is large in scale and highly dynamic, as there are up to thousands of concurrent channels, and an amateur broadcaster may start or stop a livecast session anytime; Second, the system is computationally intensive as it is preferred that raw videos uploaded from broadcasters are transcoded into multiple representations in real time for better viewer experience [4][5][6]; Third, the system is bandwidth intensive by simultaneously delivering numerous livecast channels, each may contain multiple representations, to a massive number of viewers, and both broadcasters and viewers are geographically dispersed [7][8].

Unfortunately, the computation and bandwidth resources employed by a livecast system are expensive. For example, infrastructure-as-a-service (IaaS) cloud providers like Amazon and Tencent Cloud charge dozens of US dollars per CPU core per month; and ISPs price dedicated point-to-point bandwidths thousands of dollars per Mbps per year. Given the large number of concurrent channels, it is infeasible for a livecast system to provide "full service" in all the channels, and how to allocate the limited service capacities among the channels becomes a critical problem.

Previous solutions focus on saving the system's operation cost on renting cloud instances and bandwidths by prioritizing popular channels, that is, the system transcodes videos only for popular channels, and selects video streams to replicate to edge delivery servers based on channel's local popularity [5][6][8][9]. The underlying assumption is that viewers are equally important, thus by prioritizing channels with many viewers, the system's overall benefit can be maximized. However, the assumption is valid only under the business model that most viewers watch videos for free, and the livecast website lives on advertisements. Although many websites such as Twitch initially adopt such a business model, however, a new business model that allows viewers to donate monetary virtual gifts to broadcasters, becomes increasingly popular in recent years. For example, study shows that viewers on Douyu have donated 65 million pieces of virtual gifts, which are worth over 6.8 million US dollars, in only four weeks [10]. Leading websites like Twitch and YouTube Live also adopt this business model: Twitch allows viewers to purchase "bits", a monetary virtual goods, and donate to broadcasters since 2017[1]; and YouTube Live allows viewers to purchase "super chats" with real money to highlight their comments[2]. Obviously with such a business model, a channel's *profitability*, which measures how capable it attracts monetary donations from viewers, is vitally important.

In this paper, we consider the problem that with the monetary donation business model, how to allocate service capacities among numerous channels for improving a crowdsourced livecast system's overall benefit. To understand this problem, we take Douyu as an example, dissect its *video delivery network* (*VDN*), and analyze a measurement dataset on it. We find that Douyu's system is highly dynamic with many short livecast sessions, and it follows the popularity-based service strategies in transcoding and replicating channels; moreover, the popularity and profitability of livecast channels are only moderately correlated. We also find that different genres of channels have different profitability levels, and for a livecast channel, viewers' monetary donations are closely related to the experience that they have perceived when watching the channel, therefore if more service

[1] https://help.twitch.tv/customer/portal/articles/2449458

[2] https://support.google.com/youtube/answer/7277005

capacities are allocated in a channel for improving viewers' streaming experience, more monetary donations can be expected. Our observations suggest that there exists a mismatch between the popularity-based service strategies and the business model that relies heavily on monetary donations, and channels' profitability should also be considered in allocating service capacities.

Motivated by our analysis, we present algorithms that allocate transcoding capacity among livecast channels for maximizing the system's overall benefit, which balances viewers' satisfaction with the platform's monetary profit. Our proposed proactive algorithm is optimal, and the reactive algorithm can avoid adjusting channels' transcoding schemes too often. We also present a practical strategy for VDN edge nodes to select channels to replicate, by taking channels' popularity, profitability, and bandwidth consumptions into consideration. We evaluate our proposals with experiments driven by real-world measurement data, and the results indicate that by smartly selecting channels for transcoding and replicating, the overall benefits of a crowdsourced livecast system and individual VDN edge nodes can be effectively improved.

The remainder part of this paper is organized as follows. We discuss the related works in Section 2. Section 3 introduces Douyu and gives an overview of its infrastructure. In Section 4, we present our analysis on the Douyu measurement data that motivates this work. We formulate the transcoding capacity allocating and channel replicating problems in Section 5, and present our solutions. Section 6 evaluates our proposed approaches and we conclude in Section 7.

2. Related Work

Live video streaming has been an Internet killer application for decades [11][12], and there is a rich literature on constructing, analyzing, and improving Internet IPTV systems (e.g., [2][3]). With the technological advances of cloud computing and mobile Internet, crowdsourced livecast services have emerged and become popular in recent years. Unlike the conventional IPTV services, in a crowdsourced livecast service, live videos are uploaded from many geo-distributed broadcasters, and are dispatched to viewers all over the world [8].

Since its emergence, crowdsourced livecast systems have attracted increasing attentions. Zhang *et al.* [7] examine the message and streaming latencies in livecast channels on Twitch, and study the view patterns that are influenced by both events and livecast sources. Wang *et al.* [13] investigate performances of two popular platforms, namely Periscope and Meerkat, and analyze causes of the streaming latencies. Ray *et al.* propose a live streaming upload solution for improving the overall quality of experience for viewers in crowdsourced livecast services [14]. Many works study channel popularity in terms of simultaneous viewers. Kaytoue *et al.* [15] analyze the influence of video game tournaments on channel popularity, and show that it is feasible to accurately predict a tournament channel's viewer population. Deng *et al.* [16] investigate the high churn in channel popularity that is caused by game tournaments broadcasted on Twitch. Pires *et al.* [17] study YouTube Live and Twitch, and find that on the two websites, popularity of livecast channels is more heterogeneous than what has been observed on other user-generated contents. Jia *et al.* [18] also observe that game broadcasters on Twitch have highly skewed popularity, with a significant heavy tail phenomenon.

There are relatively few works investigating the novel interactions between viewers and broadcasters. Zhang *et al.* [19] analyze the touch operations in mobile gamecasting to predict viewers' gazing patterns for smartphone energy saving. He *et al.* [20] study danmu comments in a VoD service, and use the comment volume to predict video popularity. Zhu *et al.* [21] present a statistical study on the virtual gifts received by broadcasters on Douyu. Jia *et al.* [22] analyze the social features of the videos on a YouTube-like website formed by danmu interactions. Ma *et al.* [23] present a suite of distributed algorithms for synchronizing cross-viewer community interactions. Wang *et al.* [10] present a comprehensive and insightful analysis on the interactions of posting comments and donating monetary virtual gifts on Douyu, and develop a suite of models for capturing the interaction patterns.

For improving service and reducing cost, Chen *et al.* [24] propose algorithms that allocate video transcoding server instances at geo-distributed cloud sites for saving the cloud server rental cost. Pires *et al.* [5] address the trade-off between the benefit and cost of video transcoding, and present strategies to select channels for adaptive bitrate streaming. He *et al.* [6] jointly consider video transcoding and cloud instance placement, and present algorithms for scheduling and allocating video transcoding server instances among geo-distributed cloud sites. Bilal *et al.* [9] take the latency constraints and price factors into consideration when allocating the cloud instances. Nevertheless, previous studies use viewer population to measure a channel's importance, under the assumption that viewers are equally important. In this work, we focus on livecast systems with the monetary donation business model, where donations from viewers are important revenue sources for both broadcasters and the website. We exploit channel properties that are derived from viewer interactions, especially the profitability of the channels, for improving the services. Moreover, unlike previous studies [5][6][9] that periodically execute proactive algorithms, we propose reactive algorithm that better handles the system-wide channel dynamics caused by starts and stops of the short livecast sessions, and avoids frequently adjusting the channels' transcoding schemes.

3. Background and System Overview

In this section, we give a brief introduction on Douyu. We particularly introduce the danmu-enabled viewer-broadcaster interactions, and present an overview of Douyu's infrastructure.

3.1. Background

`Douyu.com` is a leading Internet crowdsourced livecast website in China. Similar to Twitch, any registered user can set up a *channel* on Douyu and broadcast live videos on the Internet. Although initially focused on games, Douyu allows a wide range of contents to be broadcasted. For example, a *broadcaster* can sing, rap, tell stories and jokes, share experiences and opinions in her channel, or she can broadcast outdoor activities such as fishing, hiking, sightseeing to her audience. Figure 1 demonstrates viewer interface of a livecast channel, which is featured with a good-looking broadcaster playing video games.

Like many Chinese video sites, Douyu enhances video streaming with *danmu*. Danmu is a word originated from Japanese, literally meaning "bullet curtain", and it refers to a commentary sharing mechanism in which *viewers* can paste *comments* directly on top of the video while watching it. As shown in Figure 1, when

Figure 1: An example of a livecast channel on Douyu. Danmu messages are sliding from right to left, and the message starting with a space ship icon (in white circle) indicates a gift donating event; virtual gifts (in red rectangle) under different prices are displayed below the video area for viewers to purchase and donate.

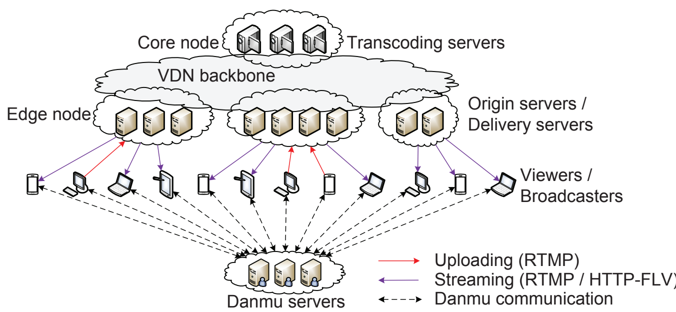


Figure 2: Overview of Douyu's infrastructure.

watching a channel, a viewer can see comments from other viewers who are watching the same channel sliding over his video screen in real time. Besides posting comments, Douyu also allows viewers to donate *monetary virtual gifts* to broadcasters, and announces the gift donations with danmu messages. Douyu provides many different virtual gifts, with prices ranging from 0.1 to thousands of RMB yuan, for viewers to purchase and donate. Danmu-enabled interactions are very popular in China, as by interacting with the broadcaster and other viewers, a viewer can feel more engaged in the broadcasted activity [10][25].

3.2. Infrastructure Overview

Douyu relies on its dedicated infrastructure to provide a nationwide service. To understand Douyu's infrastructure, we have registered a number of broadcaster and viewer accounts, upload and stream live videos to/from Douyu, monitor the communications between servers and our broadcaster/viewer clients, and analyze the traffics with a number of techniques including DNS resolution, IP-to-ASN mapping [26], and IP geolocation [27]. Note that here we do not intend to have a comprehensive measurement study on Douyu's infrastructure, but motivate further study.

Figure 2 demonstrates an overview of Douyu's infrastructure. Unlike Twitch, which deploys its servers in a few cloud data centers [8], Douyu is deployed upon a *video delivery network* (*VDN*). Logically, a VDN is composed of a *core node* and a number of geographically distributed *edge nodes*. A VDN edge node contains at least two types of servers, *origin servers* and *delivery servers*. An origin server receives raw videos uploaded from broadcasters through the RTMP protocol [28], and transfers them to the VDN core node for further processing, such as transcoding and watermarking; the delivery server replicates channel streams from the core node, and streams them to viewers using RTMP or FLV-HTTP [29]. The VDN core node resides in cloud and provides many different services, in particular, the *transcoding servers* are responsible for transcoding raw videos uploaded from broadcasters into multiple representations in various resolutions and bitrates. All VDN nodes are inter-connected with a backbone network, which provides guaranteed point-to-point bandwidths rented from the ISPs [30][31]. Douyu simultaneously employs three VDN providers, namely Wangsu[3], Tencent Cloud[4], and Dnion[5]. Among them, Wangsu is used by default, but a viewer is free switch to other providers. Besides the VDN, Douyu directly manages a set of *danmu servers*, which maintain the danmu message channels with viewers.

4. Analysis and Motivation

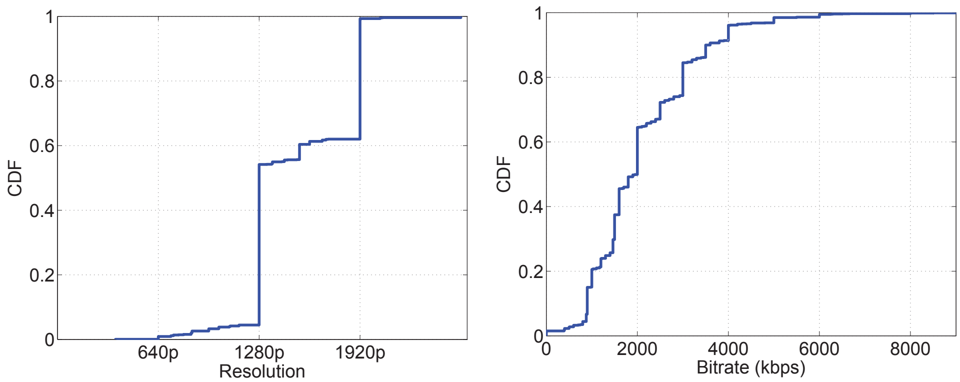
In this section, we present our measurement study on Douyu that provides insights and motivations. We first describe the measurement dataset. By mining the data, we analyze Douyu's service strategies and system dynamics, we then investigate channels' properties and study the relationship between a channel's profitability and viewers' streaming experience in the channel. Finally, we discuss our observations, and show that there exists a mismatch between Douyu's current service strategies and its business model.

4.1. Dataset

We employ a dataset from a previous measurement study on Douyu for our analysis [10][32]. By repeatedly crawling Douyu's portal webpage and webpages of the channels with on-going *livecast sessions* every five minutes, the dataset traces all the sessions in each channel, and records the channel's metadata such as channel ID, available video representations, and number of viewers. For some channels, raw video information like bitrate and resolution can also be collected. In addition, by employing a probing process to subscribe to a channel's danmu server using Douyu's REST APIs [33], we collect real-time danmu messages in all the livecast channels. In particular, two types of messages are recorded:

- **Comment message**: the message contains the comment text posted by a viewer in the channel, as well as the timestamp and ID of the comment posting viewer.
- **Gift donating message**: the message indicates a gift donating event in the channel, and contains the information such as the timestamp, the donating viewer's ID, and the gift ID. Note that we can obtain a gift's monetary price according to gift ID.

[3]https://www.wangsu.com/
[4]https://cloud.tencent.com/
[5]http://www.dnion.com/



(a) Resolution

(b) Bitrate

Figure 3: Distributions of (a) resolutions and (b) bitrates of raw videos uploaded to Douyu.

The dataset contains data in four weeks from Nov. 22 to Dec. 19, 2016, during which a total number of 1,789,027 livecast sessions were broadcasted by 242,697 distinct broadcasters[6]. By mining the danmu messages, we have identified 7,482,937 distinct viewers, who have posted 250,291,347 danmu comments, and have donated 6,894,747 pieces of virtual gifts worth 6,807,524.94 US dollars[7] in total.

4.2. Service Strategies

We first analyze Douyu's service strategies, in particular, we are interested in the strategy for allocating transcoding capacities among the channels, and the strategy for selecting channels to replicate to VDN edge nodes.

4.2.1. Transcoding strategy

Crowdsourced broadcasters usually upload raw videos of high qualities. To show this, we randomly select 800 broadcasters, and present the distributions of the resolutions and bitrates of their uploaded raw videos in over 1,700 livecast sessions in Figure 3. From the figures we can see that, most broadcasters upload raw videos with high resolutions of $1280p$ and $1920p$, and the mean video bitrate is close to 2 Mbit/s.

Unfortunately in many cases, the high-quality raw videos are not suitable to be streamed to viewers directly, as viewers using different devices (e.g., PC, pad, smartphone) prefer different resolutions (e.g., $720p$, $1080p$), and viewers under different bandwidth conditions (e.g., Ethernet, Wi-Fi, 4G/5G) prefer different bitrates. To cope with such heterogeneity, the livecast platform needs to transcode raw videos into representations with various resolutions and bitrates, so that viewers using different devices and with different bandwidths can have good experiences. Study shows that transcoding is essential for maintaining viewers' satisfaction levels in video streaming services [4].

As we have discussed in Section 1, real-time transcoding is computationally expensive, and since there are a large number of concurrent channels, it is infeasible to transcode videos for all the channels. For example,

[6]We assume that each broadcaster has a unique channel ID, and in the remaining part of this paper, we use the terms "broadcaster" and "channel" interchangeably.

[7]Virtual gifts are priced in RMB. In this paper, we use a fixed rate of 1 US dollar = 6.9 RMB yuan.

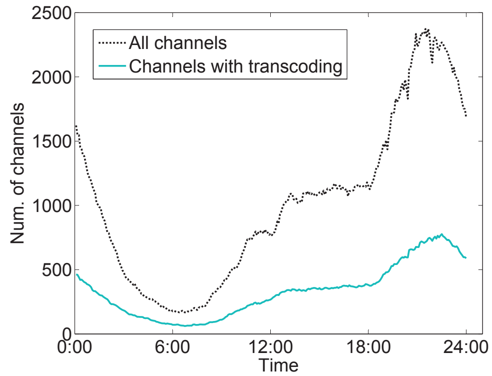
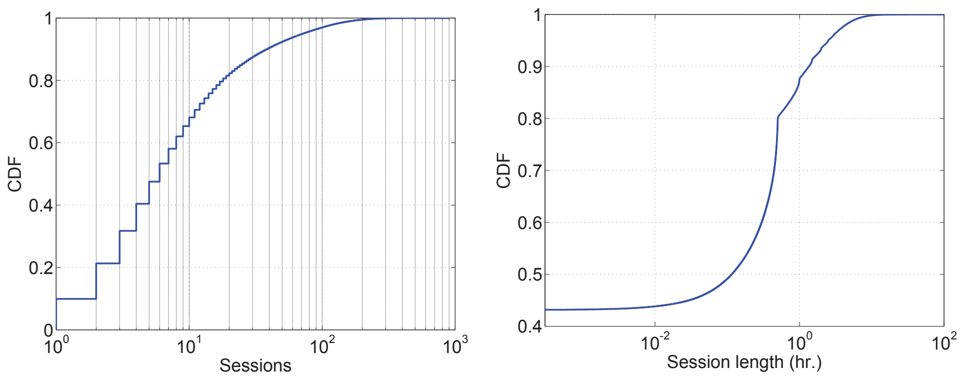


Figure 4: Concurrent channels and channels with transcoded video representations on Douyu collected every 5 minutes in Nov. 29, 2016.

it is reported that Twitch provides transcoding in only a few hundred channels that are most popular, despite that there are $10,000^+$ concurrent channels [6]. In Figure 4, we plot the concurrent livecast channels on Douyu, as well as the channels that contain transcoded video representations, in every five minutes in a day from our measurement. We can see that Douyu concurrently hosts thousands of channels in peak hours, but on average, only one third of them have two transcoded representations, targeting low and medium video qualities respectively. Further investigation tells us that Douyu only transcode videos for channels with over $15,000$ simultaneous viewers, while for the other channels, the high-quality raw videos are streamed to viewers directly, regardless of their devices and bandwidth conditions.

4.2.2. Channel replicating strategy

As discussed in Section 3, after being processed by servers in the VDN core node, livecast channels are replicated at the VDN edge nodes, so that viewers can stream from servers that are proximate to them. However, as we have seen in Figure 4, since there are thousands of concurrent channels, while the backbone connecting to an VDN edge node has limited bandwidth, it is infeasible to replicate all the channels to each edge node. A widely-adopted approach is to replicate channels based on popularity. For example, it is reported that Twitch only replicates popular channels to multiple regions, and 67% of the channels with 0 viewers are hosted in only one region [8]. We carry out an experiment on Douyu and have a similar observation. In our experiment, we create a channel on Douyu, upload live videos from a remote city, but stream (with a different viewer account) using a local computer. We repeat the experiment several times by uploading from different remote cities (using a VPN network), and each time we geolocate the origin server we upload to and the delivery server we stream from. Not surprisingly, the two servers are always at a same VDN edge node that is proximate to the broadcaster. The observation suggests that Douyu also applies the popularity-based channel replicating strategy, and our experimental channel, which is obviously among the least popular channels, can be streamed from only one edge node where it is uploaded.



(a) (b)

Figure 5: (a) Distribution of sessions initiated by broadcasters, and (b) distribution of session lengths.

4.3. System Dynamics and Channel Properties

We then investigate dynamics of the livecast channels on Douyu, and profile the channels based on the viewer-broadcaster interactions.

4.3.1. Broadcasting frequency and session length

As a crowdsourced livecast service, Douyu relies on amateur broadcasters to provide live video contents, and a broadcaster may start or stop a livecast session anytime. We consider: 1) how frequently a broadcaster initiates livecast sessions, and 2) how long a livecast session lasts.

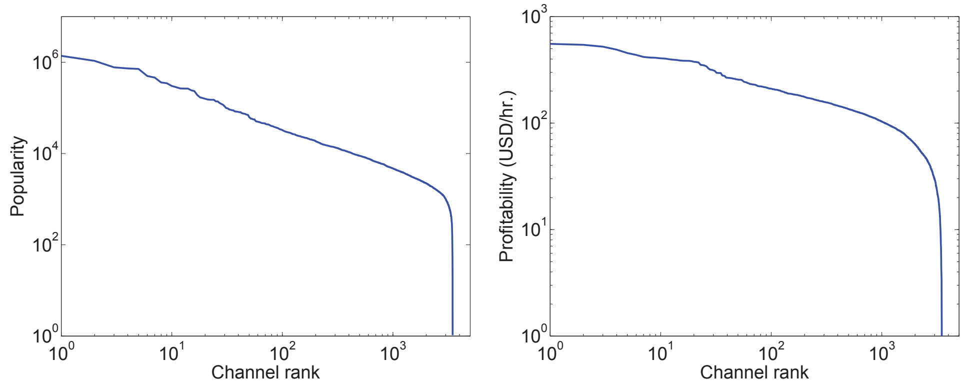
Figure 5(a) presents the distribution of the sessions initiated by the $242,697$ broadcasters in the dataset. From the figure we can see that overall the broadcasters are inactive, as half of them have no more than 6 sessions in four weeks. However, there also exist considerable active broadcasters. For example, we observe that on average, over 13% broadcasters each has at least one session per day; and for the top-10% broadcasters, averagely each has initiated over 90 sessions in four weeks.

In Figure 5(b), we present the distribution of the session lengths. We can see that most sessions do not last long: the mean session duration is 0.53 hours, and over 50% sessions last less than 10 minutes; even the top-10% longest sessions last only 3.29 hours on average. We also find that there are considerable sessions lasting exactly 30 minutes, which can be explained as Douyu rewards a broadcaster if she continuously broadcasts for over 30 minutes, therefore many broadcasters stop immediately after the threshold.

Our observations suggest that for active broadcasters, it is feasible to build profiles from their histories, as they broadcast frequently and regularly. On the other hand, most livecast sessions do not last long, therefore the livecast system is highly dynamic with broadcasters starting and terminating livecast sessions anytime, and the system needs to allocate/revoke the service capacities dynamically among the channels.

4.3.2. Popularity and profitability

We profile livecast channels based on the viewer-broadcaster interactions. For traditional TV channels, the only and most important interaction between a viewer and a channel is watching, and the property that measures how many viewers are simultaneously watching a channel is termed as the channel's *popularity*. On



(a) Popularity (b) Profitability

Figure 6: Rank distributions of the (a) popularity and (b) profitability of the 3,503 livecast channels.

the other hand, for Douyu and similar websites that allow monetary donations, another important channel property is the value of the virtual gifts that are donated during a unit broadcasting time (e.g., an hour), and we refer to such a property as the channel's *profitability*.

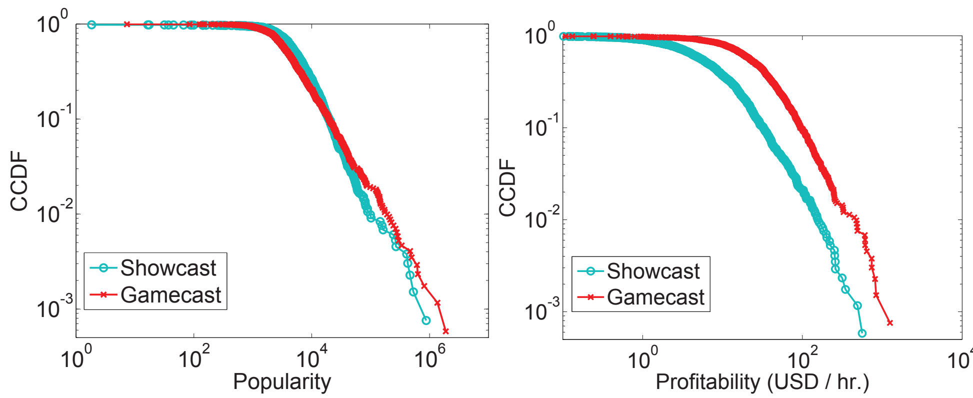
We employ the Douyu dataset to investigate the popularity and profitability of the livecast channels on Douyu. Instead of studying all the channels as in our previous work [10], here we only focus on the broadcasters that attract many viewers and gift donations, as they are indeed the candidates for allocating service capacities. More specifically, we select the top-2,500 broadcasters of the highest popularity, and the top-2,500 most profitable broadcasters from the 242,697 broadcasters. By combining them, we obtain a collection of 3,503 broadcasters. Although these broadcasters constitute only 1.44% of the broadcaster population, however, they attract as many as 87.47% monetary donations and 80.40% danmu comments. Note that Douyu provides transcoding services in all the 3,503 channels.

In Figure 6, we present the rank distributions of the popularity and profitability of the 3,503 channels. One can see that a livecast channel can be very popular and profitable: averagely there are as many as 14,261 viewers simultaneously watching a channel, and a channel receives virtual gifts worth 83.1 US dollar per hour on average. Moreover, even among the most attractive 3,503 broadcasters, they still significantly vary: For example, the top-10% most popular channels have their mean popularity 6.5 times of the averaged one from all the 3,503 channels; and the top-10% most profitable channels have their mean profitability 2.5 times of the global mean.

Since a livecast channel has two important properties, one nature question is, are they closely correlated? To answer this question, we rank the 3,503 channels according to their popularity and profitability respectively, and compute the *Spearman's rank correlation coefficient* [34] between the two ranks as

$$\rho = 1 - \frac{6\sum_{i=1}^{n}(r_{p,i} - r_{g,i})^2}{n(n^2-1)} \tag{1}$$

where n is the total number of the channels, $r_{p,i}$ and $r_{g,i}$ are the positions of channel i in the popularity and profitability ranks respectively. By applying Equation (1), we have $\rho = 0.547$. The value indicates that the two channel properties are only *moderately correlated*. In other words, a popular livecast channel on Douyu is not



(a) Popularity

(b) Profitability

Figure 7: Comparisons of the distributions of (a) popularity and (b) profitability of the gamecast and showcast channels.

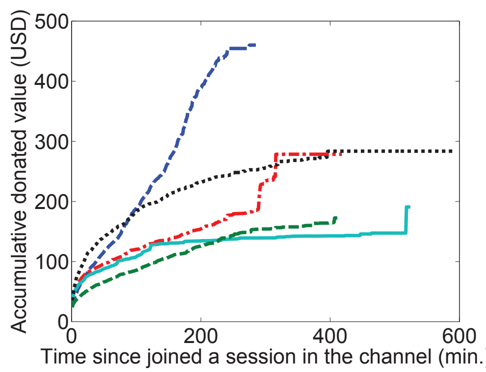


Figure 8: Accumulative gift value donated from a viewer on average since the viewer has joined, for the five channels selected from 3, 503 channels.

necessarily equally profitable, and vice versa.

4.4. Factors influencing profitability

We further investigate the factors that influence a livecast channel's profitability. Note that livecast channels on Douyu are categorized into several genres according to their contents. We first compare the profitability levels of different genres. More specifically, in the 3, 503 livecast channels, we find that 1, 716 of them are of the "gamecast" genre, and 1, 321 belong to the "showcast" genre. Figure 7 presents the popularity and profitability distributions of the gamecast and showcast channels. We can see that although the two kinds of channels have similar popularity, the showcast channels have much higher profitability levels than the gamecast ones. The observation suggests that content (including the broadcaster herself) is an important factor that influences a livecast channel's profitability.

We then analyze the relation between a viewer's watching time and his monetary donations. More specifically, for a livecast channel, we examine how much gift value a viewer has donated after he has joined the channel. We aggregate the data from all the viewers that have donated in the channel, and normalize with the viewer number. For each channel under study, we obtain a series of (*minutes*, *value*) tuples as result, in which each tuple means on average, how much money a viewer has donated since he has joined the channel for a certain period of time (in minutes). We have studied over 150 channels, and find that statistically, the longer a viewer watches the channel, the more monetary donations the broadcaster can expect to receive from him. To show this, we present the (*minutes*, *value*) series from five representative channels in Figure 8. We can see that, for all the five channels, the curves are monotonically increasing with time. The observation indicates that for improving a livecast channel's profitability, it is essential to keep the donating viewers in the channel as long as possible.

4.5. Discussion

From our analysis in this section, we can see that Douyu follows the service strategies by transcoding and replicating livecast channels based on their popularity. On the other hand, with the monetary donation business model, a channel's profitability, which measures how capable a broadcaster attracts monetary virtual gifts, is vitally important. However, our study shows that popularity and profitability of livecast channels are only moderately correlated.

Meanwhile, it is well known that good video streaming experience will improve viewer engagement. For example, previous study reports that in live video streaming, a viewer's watching time is monotonically increasing with the highest bitrate he can stream [35]. In a crowdsourced livecast service, when a channel is allocated with more transcoding capacity, it can provide more representations, and a viewer can select a representation that best suites his bandwidth condition; similarly, replicating a channel to edge nodes that are close to viewers will also improve their streaming experience. Since better streaming experience leads to longer watching time, and according to our observation, the longer a donating viewer watches the channel, the more monetary gifts he will donate, we can conclude that *by allocating more service capacities to a livecast channel, more monetary donations from viewers can be expected.*

The above analysis suggests that there exists a *mismatch* between Douyu's current service strategies and its business model: with the popularity-based strategies, some profitable but not equally popular livecast channels will not be allocated with sufficient service capacities. As a result, in these channels, viewers may have few video representation choices, or are forced to stream from remote delivery servers. As a consequence, suffering poor streaming experience, viewers will have a shorter channel watching time, and donate less virtual gifts comparing with the case if the channels are allocated with sufficient service capacities.

5. Improving Crowdsourced Livecast Service

Inspired by the insights from our measurement study, we believe that in addition to livecast channels' popularity, profitability should also be considered in a Douyu-like livecast system's service strategies. In this section, we address the problems of allocating transcoding and delivery capacities among numerous channels, and present our solutions. To make our approaches more tractable, we restrict our discussion on Douyu-like platforms.

5.1. *Allocating Video Transcoding Capacity*

5.1.1. *Problem formulation*

As discussed in Section 4.2, since it is infeasible for a crowdsourced livecast system to transcode videos in all the channels, then for each channel, the system must decide whether its video should be transcoded or not, and if yes, how many representations should be transcoded?

We formulate the transcoding capacity allocating problem as the following. We suppose that the livecast system supports a set of m representations, denoted as $\mathcal{R}$. We use Ω_j to denote all the possible representation sets that each set is a subset of $\mathcal{R}$, and contains j representations. For example, suppose a channel's raw video representation is $1280p$, and the system is capable to transcode it into two representations as $720p$ and $1080p$, then we have $\Omega_1 = \{\{1280p\}\}$, $\Omega_2 = \{\{720p, 1080p\}, \{720p, 1280p\}, \{1080p, 1280p\}\}$, and $\Omega_3 = \{\{720p, 1080p, 1280p\}\}$. Here we assume that if only one representation is provided to viewers, it is the raw video's representation, as observed in Section 4.2.

For each channel, say channel i, the livecast system applies a transcoding scheme w_i to it, whose value is a subset of $\mathcal{R}$, and if the set contains j representations, we say $w_i \in \Omega_j$. For example, if $w_i \in \Omega_1$, it means that the raw video is directly streamed to viewers without transcoding; while $w_i \in \Omega_m$ indicates that in addition to the raw video, the channel also contains all the representations that the system supports (i.e., full transcoding).

Different transcoding schemes bring different levels of user experience quality. In this work, we follow the method in [4][6] to model the relationship between a transcoding scheme w_i and its streaming experience as

$$s(w_i) = a \cdot log(|w_i|) + b \tag{2}$$

where the constant a and b are chosen so that if $w_i \in \Omega_m$, $s(w_i) = 1$, that is, full transcoding leads to the best streaming experience.

Applying a transcoding scheme w_i on channel i consumes transcoding capacity and incurs cost. Since the VDN core node is deployed on a cloud, we use the cost of renting server instances as the transcoding cost. Specifically, we use $c(w_i)$ to denote the per-hour rental cost associated with the transcoding scheme w_i, and clearly if $w_i \in \Omega_1$, we have $c(w_i) = 0$.

For each livecast channel, the system traces two of its properties, i.e., *popularity* and *profitability*: A channel, say channel i's popularity p_i, is defined as the number of its simultaneous viewers in a recent period of time, and its profitability g_i is the per-hour value of the virtual gifts donated by the viewers recently. Clearly, the higher the g_i/p_i ratio is, the more generous the viewers are in donating virtual gifts in the channel.

If a transcoding scheme w_i is applied to channel i, then we can compute the channel's *satisfaction* as

$$S_i(w_i) = p_i \cdot s(w_i) \tag{3}$$

and its *profit*, which is the income from the received monetary donations minus the transcoding cost, can be computed as

$$P_i(w_i) = \Gamma(s(w_i)) \cdot g_i - c(w_i) \tag{4}$$

where $\Gamma(\cdot)$ is the function that reflects the relationship between viewers' streaming experience and their willingness to continue to watch the channel and donate. In general, $\Gamma(s(w_i))$ should be monotonically increasing with $s(w_i)$, as observed in Section 4.4, better streaming experience leads to more donations from viewers.

Combining $S_i(w_i)$ and $P_i(w_i)$, we can define channel i's *gain* at a transcoding scheme w_i as

$$G_i(w_i) = \alpha \cdot S_i(w_i) + \beta \cdot P_i(w_i) \tag{5}$$

where α and β are weight parameters with $\alpha + \beta = 1$, and their values reflect how the system balances viewers' satisfaction with monetary profit.

With the above formulation, we formulate the *transcoding capacity allocating problem* as: for each livecast channel i with on-going sessions, decide a transcoding scheme w_i, so as to maximize the system's *overall gain* as $\mathbf{G}(w_1, \cdots, w_n) = \sum_i^n G_i(w_i)$, under the constraint of the system's available transcoding capacity C, i.e.,

$$\begin{array}{ll} \text{Maximize} & \mathbf{G}(w_1, \cdots, w_n) = \sum_{i=1}^n G_i(w_i) \\ \text{s.t.} & \sum_{i=1}^n c(w_i) \le C \end{array} \tag{6}$$

Note that for a cloud-based VPN core node, its capacity C should be right-sized with the diurnal workload. However, in this work we do not address the capacity right-sizing problem, as it is well studied [36]. For simplicity, C is assumed as a constant.

5.1.2. Proactive Transcoding Capacity Allocating

Before presenting our solution, we first introduce some notations. For a livecast channel i, suppose its current transcoding scheme is $w_i \in \Omega_j$, where $j = 0, 1, \cdots, m$, we denote w_i^+ as a transcoding scheme if it contains one more representation than w_i, i.e., $w_i^+ \in \Omega_{j+1}$, if $j = 0, 1, \cdots, m-1$; or $w_i^+ = w_i$, if $j = m$. And we denote w_i^- as the transcoding scheme if one representation is removed from w_i, that is, $w_i^- \in \Omega_{j-1}$ if $j = 1, \cdots, m$, or $w_i^- = w_i$, if $j = 0$.

With the notations, we define $\Delta G_i^+(w_i) = G_i(w_i^+) - G_i(w_i)$ or $\Delta G_i^-(w_i) = G_i(w_i^-) - G_i(w_i)$ as the differences of channel i's gain if its transcoding scheme is promoted from w_i to w_i^+ or demoted from w_i to w_i^- respectively.

We first present a *proactive transcoding capacity allocating algorithm* (*P-TCAA*) for solving the problem. As presented in Algorithm 1, the algorithm iteratively executes, and in each iteration, it allocates the transcoding capacity among all the channels with on-going sessions. Initially, the algorithm temporarily sets all the channels' transcoding schemes as $w_i \in \Omega_1$ and revokes the capacity (line 1-2). Then iteratively, the algorithm compares the gain increases $\Delta G_i^+(w_i)$ for all the channels given that each channel's transcoding scheme is hypothetically promoted to w_i^+; it selects channel x that has the largest gain increase, promotes its transcoding scheme to w_x^+, and updates the system's spare capacity (line 5-9). The iteration aggressively promotes channels' transcoding schemes until no spare capacity can be further allocated (line 11).

Theorem 1. *The P-TCAA algorithm is optimal.*

Proof. Suppose that the system has a capacity for transcoding C video representations in real time. We prove by induction that after C iterations, the algorithm achieves the maximum overall gain $\mathbf{G}$.

Algorithm 1 Proactive transcoding capacity allocating algorithm (P-TCAA)

for each livecast channel i **do**
    $w_i \in \Omega_1$; ▷ Initialize transcoding schemes
end for
repeat
    Sort $\Delta G_i^+(w_i)$ in descending order for all channels;
    Let x be the first channel in the rank;
    if $(\Delta G_x^+(w_x) > 0)$ **then**
        $w_x \leftarrow w_x^+$;
        Update $\Delta G_x^+(w_x)$ and the spare capacity;
    end if
until no spare transcoding capacity can be allocated

For the basic case of $C = 1$, clearly the algorithm is optimal by selecting the channel with the largest gain increase $\Delta G_i^+(w_i)$. Now assume that after the k^{th} iteration, a maximum system gain $\mathbf{G}(w_1, \cdots, w_n)$ is achieved. For the $(k+1)^{th}$ iteration, suppose that the algorithm selects channel x to promote, i.e., $w_x \leftarrow w_x^+$. If the resulting overall gain $\mathbf{G}(w_1, \cdots, w_x^+, \cdots w_n)$ is not maximal, then there exists a channel y such that by promoting w_y to w_y^+, the maximum system gain $\mathbf{G}(w_1, \cdots, w_y^+, \cdots, w_n)$ can be achieved. Recall that $\mathbf{G}(w_1, \cdots, w_x^+, \cdots w_n) = \mathbf{G}(w_1, \cdots, w_n) + \Delta G_x^+(w_x)$, clearly if $\mathbf{G}(w_1, \cdots, w_y^+, \cdots, w_n) > \mathbf{G}(w_1, \cdots, w_x^+, \cdots, w_n)$, we shall have $\Delta G_y^+(w_y) > \Delta G_x^+(w_x)$, which contradicts the fact the x ranks first among all the channels regarding $G_i^+(w_i)$. Therefore, channel y doesn't exist, and $\mathbf{G}(w_1, \cdots, w_x^+, \cdots w_n)$ is the maximum overall gain of the livecast system after the $(k+1)^{th}$ iteration. □

5.1.3. Reactive Transcoding Capacity Allocating

One drawback of the periodically executed proactive algorithm is that when a channel starts to broadcast after an algorithm execution, it doesn't have any transcoding capacity allocated, until next time the algorithm executes. However, from Section 4.3 we can see that most livecast sessions do not last long, thus if the algorithm's execution interval is too long, some channels with short sessions may never have transcoding capacity allocated. To overcome this problem, we present the *reactive transcoding capacity allocating algorithm* (*R-TCAA*), as presented in Algorithm 2.

The R-TCAA algorithm is invoked on the event when a broadcaster x initiates a new livecast session. The algorithm first examines whether there is spare transcoding capacity, if yes, the algorithm behaves similarly to P-TCAA: it compares all the channels' gain increases $\Delta G_i^+(w_i)$ and selects the one with the largest increase to promote; the iteration continues until no capacity can be further allocated (line 2-10).

When there is no spare capacity, the algorithm seeks to demote some existing channels' transcoding schemes, revokes the capacity, and allocates to the new channel x. To find the channels for demoting, the algorithm compares all the channels' gain decreases $\Delta G_i^-(w_i)$, and selects the one with the smallest decrease to demote (line 13-18). The iteration repeats until the system's overall gain G can not be further improved (line 15 and

Algorithm 2 Reactive transcoding capacity allocating algorithm (R-TCAA)

Input: New livecast channel x;

$w_x \in \Omega_1$;
if there is spare transcoding capacity **then**
  repeat
    Sort $\Delta G_i^+(w_i)$ in descending order for all channels and channel x;
    Let y be the first channel in the rank;
    if $(\Delta G_y^+(w_y) > 0)$ **then**
      $w_y \leftarrow w_y^+$;
      Update $\Delta G_y^+(w_y)$ and the spare capacity;
    end if
  until no spare transcoding capacity can be allocated
else
  repeat
    Sort $\Delta G_i^-(w_i)$ in descending order for all channels;
    Let y be the first channel in the rank;
    if $(\Delta G_x^+(w_x) + \Delta G_y^-(w_y) > 0)$ **then**
      $w_x \leftarrow w_x^+$; $w_y \leftarrow w_y^-$;
      Update $\Delta G_x^+(w_x)$, $\Delta G_x^-(w_x)$, $\Delta G_y^+(w_y)$, and $\Delta G_y^-(w_y)$;
    end if
  until no adjustment can be performed
end if

19).

One consequence of Algorithm 2 is that it may adjust a channel's transcoding scheme too often. To overcome this problem, we amend the algorithm as the following: each time a channel is selected for promoting or demoting, we examine the time of its last transcoding scheme adjustment, and skip the channel if its last adjustment happened no earlier than a threshold τ.

5.2. Channel Replicating Strategy

We also consider the channel replicating problem, that is, among the numerous livecast channels, which channels a VDN edge node should select to replicate, with its limited backbone bandwidth B connecting to the core node.

One straightforward way is to select the top-N channels with the most local viewers under the constraint of $\sum_{i=1}^{N} b(w_i) \leq B$, where $b(w_i)$ is the bandwidth required to receive all video representations in channel i under its transcoding scheme w_i. Note that Twitch and Douyu currently replicate channels based on popularity [8]. However, as we have analyzed in Section 4, on a Douyu-like platform, a channel's profitability should also be considered in making the replicating decisions.

Our key idea is to compute a local *utility* for each candidate channel, and select the top-N channels with the highest *utility-bandwidth ratios*. More specifically, for a candidate channel i, the edge node traces its *local popularity* $p_l(i)$ as the portion of the channel's viewers in the node's region divided by the local viewers in all the channels. Similarly, the node also traces the channel's *local profitability* $r_l(i)$ as the ratio of the gift value from the local viewers in channel i divided by the gift value donated by all the local viewers. With the two metrics, the edge node computes channel i's *local utility* as

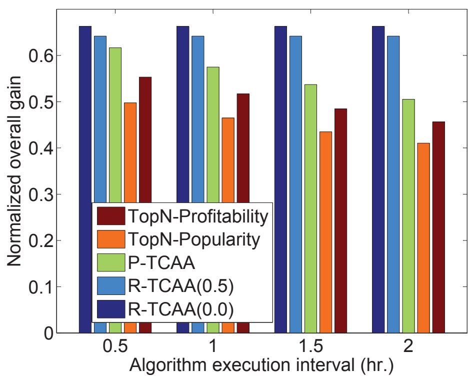
$$u_l(i) = c_1 \cdot p_l(i) + c_2 \cdot r_l(i) \tag{7}$$

where c_1 and c_2 are weight parameters with $c_1+c_2 = 1$. The VDN edge node computes the *local utility-bandwidth ratio* for channel i as

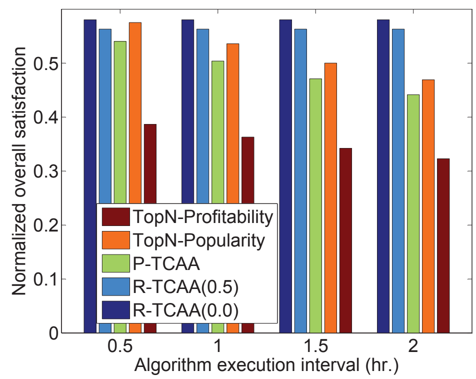
$$o(i) = \frac{u_l(i)}{b(w_i)} \tag{8}$$

and it selects the top-N channels with the highest ratios until there is no spare backbone bandwidth.

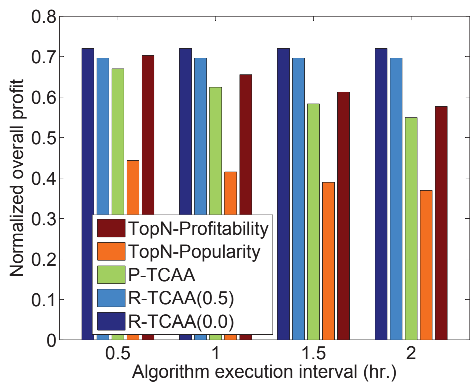
Finally, we want to point out that although we focus on a Douyu-like system, however the general principle of balancing both popularity and profitability of channels in allocating service capacities, is applicable to other system contexts. For example in Twitch, which places its servers in a few cloud data centers in different countries [8], by taking the factors such as various bandwidth and server rental prices across different data center locations into consideration, we can formulate a problem for maximizing the system's overall benefit, like the one in Equation (6), and such a problem can be solved with similar techniques as we have used in Algorithm 1 and Algorithm 2.



(a) Overall gain



(b) Overall satisfaction



(c) Overall profit

Figure 9: (a) Overall gain, (b) overall satisfaction, and (c) overall profit of simulated livecast system under different transcoding capacity allocating algorithms with the proactive algorithms' execution intervals varying from 0.5 to 2.0 hours.

6. Evaluation

6.1. Experiment Setup

We simulate a crowdsourced livecast system for evaluating our proposed solutions. The simulation experiments are driven by the data from our real-world measurement on Douyu. More specifically, in the experiments, a channel's session length follows the distribution as in Figure 5(b), and we control the Poisson session join rate so that on average, there are $2,000$ active channels in the simulated system. For each livecast channel, we associate it to a random channel from the Douyu dataset, so that the simulated channel has a real-world channel's popularity and profitability, and for all the simulated channels, the two properties have a Spearman's rank correlation coefficient of 0.547. We assume that transcoding one video representation in real time requires one unit capacity (e.g., one CPU core), and consult the rental price of Tencent Cloud[8] for the transcoding cost.

For the function $\Gamma(\cdot)$ in Equation (4), which reflects the relation between viewers' streaming experience in the channel and the their willingness to continue to watch and donate, we temporarily set $\Gamma(s(w)) = (s(w))^{\gamma}$, with $\gamma = 1$ in our simulation. We recognize that it is challenging to find a function that accurately describes the relation, and leave it for our future work.

The other parameters are set as the following. The simulated livecast system supports up to $|\mathcal{R}| = 5$ video representations. We let $b = 0.4$ in Equation (2), so that even under no transcoding, viewers still have a 40% streaming satisfaction. We set the parameters α and β in Equation (5) as 0.5, and let $c_1 = c_2 = 0.5$ in Equation (7).

6.2. Evaluating Transcoding Capacity Allocating Algorithms

We consider the transcoding capacity allocating problem at a VDN core node that is capable to transcode $C = 1,000$ video representations in real time. We evaluate and compare the following algorithms.

- *P-TCAA*: The algorithm is presented in Algorithm 1, and if not otherwise specified, the algorithm's periodical execution interval is set as one hour.

- *R-TCAA*: The algorithm is presented in Algorithm 2. We denote the R-TCAA algorithm with a channel adjusting interval threshold τ as R-TCAA(τ), and by default, τ is set as 0.5 hours.

- *Top$N$-Popularity*: We refer Top$N$-Popularity to the case that only channels' popularity is considered when applying the P-TCAA algorithm, by letting $\alpha = 1.0$ and $\beta = 0.0$ in Equation (5). Note that Twitch and Douyu allocate transcoding capacity based on popularity only, as analyzed in Section 4.

- *Top$N$-Profitability*: We refer Top$N$-Profitability to the case that only channels' profitability is considered when running the P-TCAA algorithm, by letting $\alpha = 0.0$ and $\beta = 1.0$ in Equation (5), and for the Top$N$-Popularity and Top$N$-Profitability algorithms, their default execution intervals are one hour.

[8]https://buy.cloud.tencent.com/price/cvm/calculator

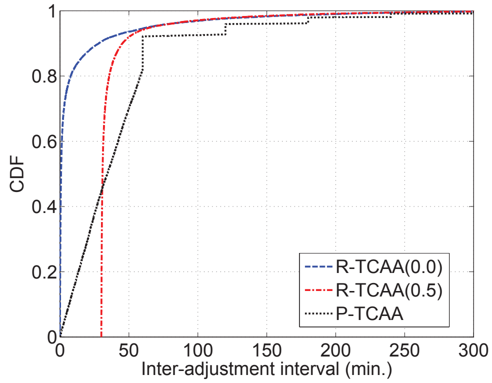


Figure 10: Distributions of channels' transcoding scheme adjustment intervals under different algorithms.

We examine the overall gain $\mathbf{G}$ of the simulated livecast system as defined in Equation (6). We also evaluate the viewers' overall satisfaction and the system's overall profit, which are sums of the individual channel's satisfaction and profit as in Equation (3) and Equation (4) respectively.

6.2.1. Overall performance

We first have an overall evaluation on different capacity allocating algorithms. Since the algorithms of P-TCAA, Top$N$-Popularity, and Top$N$-Profitability are periodically executed, we vary the algorithms' execution intervals from 0.5 to 2.0 hours. And for the R-TCAA algorithm, we consider the case that there is no constraint for adjusting a channel's transcoding scheme (i.e., R-TCAA(0.0)), and the case that the constraint threshold τ is 0.5 hours (i.e., R-TCAA(0.5)).

Figure 9 presents the overall gain, overall satisfaction, and overall profit of the livecast system under different algorithms. All the metrics are normalized against the ones that are achieved by R-TCAA(0.0) with unlimited transcoding capacity (i.e., $C = \infty$). From the figures we can make the following observations: First, among the proactive algorithms, P-TCAA outperforms Top$N$-Popularity and Top$N$-Profitability, while Top$N$-Popularity or Top$N$-Profitability only has good performance in satisfaction or profit. Second, the three proactive algorithms have better performances when they are executed more frequently, while the reactive R-TCAA algorithm has superior performances comparing with the proactive ones, suggesting that R-TCAA better handles the system-wide channel dynamics caused by the starts and stops of livecast sessions. Finally, when imposing a channel adjustment interval constraint of $\tau = 0.5$ hours (i.e., R-TCAA(0.5)), the system's overall gain is only slightly reduced, comparing with the case that no constraint is imposed (i.e., R-TCAA(0.0)).

6.2.2. Transcoding scheme adjustment frequency

Besides the overall gain, we also examine the intervals between an individual channel's consecutive adjustments on its transcoding scheme during a session, since if a channel's transcoding scheme is adjusted too often, viewers may feel upset by having to re-select the representation frequently.

Figure 10 plots the distributions of the channels' transcoding scheme adjustment intervals under R-TCAA(0.0), R-TCAA(0.5), and P-TCAA. From the figure we can see that, when no constraint is imposed on the reactive R-TCAA algorithm, many channels have their transcoding schemes adjusted very frequently, as over 80% of the

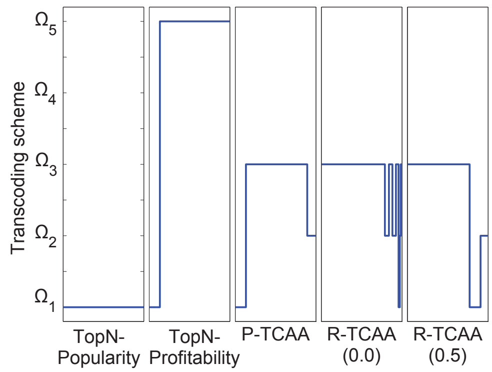


Figure 11: Adjustments of an exemplary channel's transcoding scheme within a 236-minute session under different algorithms.

inter-adjustment intervals are less than 10 minutes; nevertheless, when a constraint threshold of $\tau = 0.5$ hours is imposed, such frequent adjustments are avoided, while the system's overall gain is only slightly reduced, as indicated in Figure 9(a). For the proactive P-TCAA algorithm, it is observed that over 90% of the channels have their transcoding schemes adjusted on every algorithm execution, suggesting that the proactive algorithm, which adjust channels' transcoding schemes periodically, can not respond to the channel dynamics in time.

6.2.3. Case study

To better understand the different transcoding capacity allocating algorithms, we present evolution of an exemplary channel's transcoding scheme during a 236-minute livecast session under different algorithms in Figure 11. We can see that under the Top$N$-Popularity algorithm, the channel's transcoding scheme stays at Ω_1 without any capacity allocated all the time, but with Top$N$-Profitability, the channel is assigned with the full transcoding scheme Ω_5 at the first time the algorithm is executed, due to the channel's high profitability but relatively low popularity. The P-TCAA and R-TCAA algorithms set the channel's transcoding scheme at Ω_3 most of the time, which is more reasonable. However, under P-TCAA, the channel has no capacity allocated for over 30 minutes, while R-TCAA(0.0) adjusts the channel's transcoding scheme as many as 7 times. Only under R-TCAA(0.5), the channel is assigned with the transcoding scheme Ω_3 immediately after the session starts, does not have its transcoding scheme frequently adjusted, but is still sensitive enough to the system-wide channel dynamics by having its transcoding scheme adjusted to Ω_1 and Ω_2 at the 183^{rd} and 215^{th} minute respectively.

6.2.4. Influence of available transcoding capacity

In the previous experiments, we assume that the VDN core code is capable to transcode $C = 1,000$ video representations simultaneously. In this experiment, we vary the system's transcoding capacity C, and investigate how it influences the transcoding capacity allocating algorithms.

Figure 12 presents the simulated livecast system's overall gains achieved by different algorithms under various C. We can see that the reactive algorithms outperform proactive ones all the time. More importantly, we find that the performance is not linearly improved with the available transcoding capacity; and for all the algorithms, after the capacity C exceeds $5,000$, the improvements are trivial. We explain the observation with

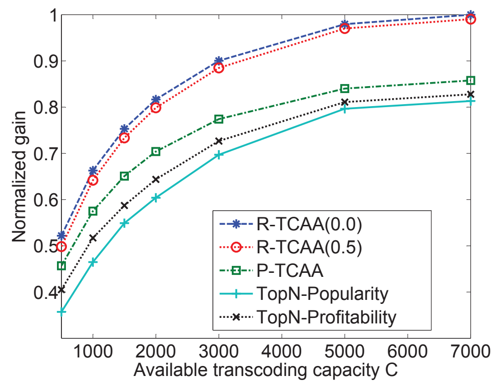


Figure 12: Influences of available transcoding capacity on different algorithms.

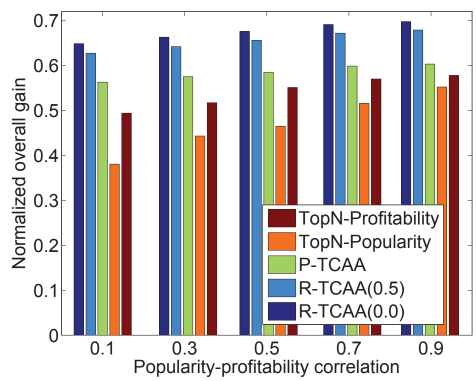


Figure 13: Influences of channels' popularity-profitability rank correlation on different algorithms.

the fact that although transcoding more video representations in channels can improve viewers' satisfaction levels and their willingness to donate, however, it also incurs cost for renting cloud server instances. For some channels with low popularity and profitability, the benefit of having more video representations doesn't compensate for the cost, and the algorithms do not allocate transcoding capacities to these channels.

6.2.5. Influence of popularity-profitability correlation

In the previous experiments, livecast channels' popularity and profitability are only moderately correlated, with a Spearman's rank correlation coefficient of 0.547, as observed from our measurement study. In this experiment, we investigate the impact of the channels' popularity-profitability correlation.

Our simulation requires livecast channels to have a desired popularity-profitability correlation. More specifically, we seek to synthesize n livecast channels with their popularity and profitability following the distributions as in Figure 6(a) and Figure 6(b) respectively, but their popularity-profitability rank correlation coefficient ρ can be arbitrarily given.

We use the following method to generate the desired channel data [37]. We first select two random variables i_1 and i_2 from $[1, \cdots, n]$, and compute

$$j_1 = \rho \times i_1 + \sqrt{(1-\rho^2)} \times i_2, \tag{9}$$

where ρ is the desired rank correlation coefficient. We then create a channel x, and assign its popularity and

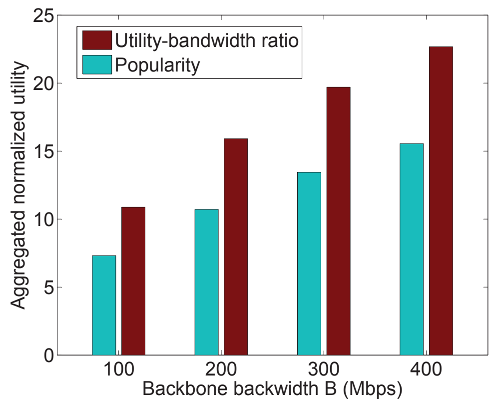


Figure 14: Comparison of utility-bandwidth ratio-based channel replication strategy and popularity-based strategy under varying backbone bandwidth B.

profitability with the i_1^{th} and j_1^{th} highest values in the popularity and profitability ranks from the measurement data respectively. By repeating these steps, we can create a set of livecast channels, and statistically, their Spearman's rank correlation coefficient between popularity and profitability is ρ.

We synthesize five groups of livecast channels, each containing $5,000$ channels, and their popularity-profitability rank correlation coefficients are 0.1, 0.3, 0.5, 0.7, and 0.9 respectively. For each group, we apply the R-TCAA, P-TCAA, Top$N$-Popularity, and Top$N$-Profitability algorithms to allocate transcoding capacity. Figure 13 presents the experiment results, from which we can see that in general, the system's overall gain is reduced when the correlation between the channels' popularity and profitability is weak, especially for the Top$N$ algorithms, as only one channel property is considered; on the other hand, the algorithms of P-TCAA and R-TCAA are less influenced, as in these algorithms, both the viewers' satisfaction and the monetary profit are taken into consideration.

6.3. Evaluating Channel Replicating Strategy

We evaluate our proposed channel replicating strategy in this section. As in our previous experiments, we use the channels' popularity and profitability data as collected from Douyu to drive the simulation. We assume that the bandwidth consumptions of the five video representations are 400 kbit/s, 600 kbit/s, 850 kbit/s, 1150 kbit/s, and 1500 kbit/s respectively, and the system employs the R-TCAA algorithm with $\tau = 0.5$ hours to allocate the transcoding capacity among the channels. We consider a VDN edge node with a backbone bandwidth of B, which we vary in the experiment.

In Figure 14, we present the aggregated utility of the channels that are replicated by the edge node using the strategy based on utility-bandwidth ratio as proposed in Section 5.2, and we also plot the aggregated utility achieved by the popularity-based strategy for comparison. For both strategies, we vary the backbone bandwidth B form 100 Mbit/s to 400 Mbit/s. From the figure one can see that our proposed strategy obviously outperforms the popularity-based one. The better performance can be explained as in our proposed strategy, a channel's popularity is combined with its profitability when being considered for replicating. Moreover, we use the utility-bandwidth ratio as the criteria, thus can avoid replicating the channels that have only moderate utilities in the local region but require much bandwidth with many video representations.

7. Conclusion

In this paper, we sought to understand and improve Internet crowdsourced live video broadcasting (livecast) services. By analyzing real-world measurement data from Douyu, the leading crowdsourced livecast website in China, we found that its system is deployed upon a video delivery network (VDN), and how to allocate limited service capacities, including the transcoding and delivery capacities, among the numerous channels becomes a critical problem. Moreover, we found that Douyu follows the strategies that prioritize popular channels in terms of simultaneous viewers in allocating service capacities, and there is a mismatch between such service strategies and the website's business model that heavily relies on monetary donations from viewers.

Motivated by our analysis, we proposed that channels' profitability, which measures how capable a channel attracts monetary donations from viewers, should be considered in allocating service capacities. We presented algorithms that balance viewers' satisfaction with the system's monetary profit in allocating transcoding capacity among livecast channels, and we also proposed a practical strategy for VDN edge nodes to select channels to replicate. The experiments driven by real-world measurement data indicated that our proposed approaches can effectively improve the overall benefits of a livecast system and individual VDN edge nodes.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grants 61672486 and 61672485, and in part by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDC02070300.

References

[1] M. DiPietro, Twitch is 4th in peak US Internet traffic (Feb. 2014).
URL `http://blog.twitch.tv/twitch-is-4th-in-peak-us-internet-traffic-90b1295af358`

[2] H. Yin, X. Liu, T. Zhan, V. Sekar, F. Qiu, C. Lin, H. Zhang, B. Li, Design and deployment of a hybrid CDN-P2P system for live video streaming: Experiences with LiveSky, in: Proc. of MM'09, Beijing, China, 2009, pp. 25–34.

[3] S. Zou, Q. Wang, J. Ge, Y. Tian, Peer-assisted video streaming with RTMFP Flash player: A measurement study on PPTV, IEEE Trans. on Circuits and Systems for Video Technology 28 (1) (2018) 158–170.

[4] L. Toni, R. Aparicio-Pardo, K. Pires, G. Simon, A. Blanc, P. Frossard, Optimal selection of adaptive streaming representations, ACM Trans. on Multimedia Computing, Communications, and Applications 11 (2s) (2015) 43.

[5] K. Pires, G. Simon, DASH in Twitch: Adaptive bitrate streaming in live game streaming platforms, in: Proc. of Workshop on Design, Quality and Deployment of Adaptive Video Streaming (VideoNext'14), Sydney, Australia, 2014, pp. 13–18.

[6] Q. He, J. Liu, C. Wang, B. Li, Coping with heterogeneous video contributors and viewers in crowdsourced live streaming: A cloud-based approach, IEEE Trans. on Multimedia 18 (5) (2016) 916–928.

[7] C. Zhang, J. Liu, On crowdsourced interactive live streaming: A Twitch.TV-based measurement study, in: Proc. of ACM Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'15), Portland, OR, USA, 2015, pp. 55–60.

[8] J. Deng, G. Tyson, F. Cuadrado, S. Uhlig, Internet scale user-generated live video streaming: The twitch case, in: Proc. of Passive and Active Measurement Conference (PAM'17), Sydney, Australia, 2017, pp. 60–71.

[9] K. Bilal, A. Erbad, M. Hefeeda, QoE-aware distributed cloud-based live streaming of multisourced multiview videos, J. of Network and Computer Applications 120 (2018) 133–144.

[10] X. Wang, Y. Tian, R. Lan, W. Yang, X. Zhang, Beyond the watching: Understanding viewer interactions in crowdsourced live video broadcasting services, IEEE Trans. on Circuits and Systems for Video Technology 29 (11) (2019) 43.

[11] B. Li, Z. Wang, J. Liu, W. Zhu, Two decades of Internet video streaming: A retrospective view, ACM Trans. on Multimedia Computing, Communications, and Applications 9 (1s) (2013) 33.

[12] M. Wang, C. Xu, S. Jia, G.-M. Muntean, Video streaming distribution over mobile Internet: A survey, Frontiers of Computer Science 12 (6) (2018) 1039–1059.

[13] B. Wang, X. Zhang, G. Wang, H. Zheng, B. Y. Zhao, Anatomy of a personalized livestreaming system, in: Proc. of Internet Measurement Conference (IMC'16), Santa Monica, CA, USA, 2016, pp. 485–498.

[14] D. Ray, J. Kosaian, K. V. Rashmi, S. Seshan, Vantage: optimizing video upload for time-shifted viewing of social live streams, in: Proc. of ACM SIGCOMM'19, Beijing, China, 2019, pp. 380–393.

[15] M. Kaytoue, A. Silva, L. Cerf, W. Meira, C. Raïssi, Watch me playing, I am a professional: A first study on video game live streaming, in: Proc. of WWW'12 Companion, Lyon, France, 2012, pp. 1181–1188.

[16] J. Deng, F. Cuadrado, G. Tyson, S. Uhlig, Behind the game: Exploring the Twitch streaming platform, in: Proc. of International Workshop on Network and Systems Support for Games (NetGames'15), Zagreb, Croatia, 2015, pp. 1–6.

[17] K. Pires, G. Simon, YouTube Live and Twitch: A tour of user-generated live streaming systems, in: Proc. of Proc. of ACM Multimedia Systems (MMSys'15), Portland, OR, USA, 2015, pp. 225–230.

[18] A. L. Jia, S. Shen, D. Epema, A. Iosup, When game becomes life: The creators and spectators of online game replays and live streaming, ACM Trans. on Multimedia Computing, Communications, and Applications 12 (4) (2016) 47.

[19] C. Zhang, Q. He, J. Liu, Z. Wang, Beyond the touch: Interaction-aware mobile gamecasting with gazing pattern prediction, in: Proc. of IEEE INFOCOM'17, Atlanta, GA, USA, 2017, pp. 1–9.

[20] M. He, Y. Ge, L. Wu, E. Chen, C. Tan, Predicting the popularity of danmu-enabled videos: A multi-factor view, in: Proc. of International Conference on Database Systems for Advanced Applications (DASFAA'16), LNCS 9643, Dallas, TX, USA, 2016, pp. 351–366.

[21] Z. Zhu, Z. Yang, Y. Dai, Understanding the gift-sending interaction on live-streaming video websites, in: Proc. of International Conference on Social Computing and Social Media, Vancouver, BC, Canada, 2017, pp. 274–285.

[22] A. L. Jia, S. Shen, D. Li, S. Chen, Predicting the implicit and the explicit video popularity in a user generated content site with enhanced social features, Computer Networks 140 (2018) 112–125.

[23] X. Ma, C. Zhang, J. Liu, R. Shea, D. Fu, Live broadcast with community interactions: Bottlenecks and optimizations, IEEE Trans. on Multimedia 19 (6) (2017) 1184–1194.

[24] F. Chen, C. Zhang, F. Wang, J. Liu, X. Wang, Y. Liu, Cloud-assisted live streaming for crowdsourced multimedia content, IEEE Trans. on Multimedia 17 (9) (2015) 1471–1483.

[25] O. L. Haimson, J. C. Tang, What makes live events engaging on Facebook Live, Periscope, and Snapchat, in: Proc. of CHI'17, Denver, CO, USA, 2017, pp. 48–60.

[26] Cymru, Team Cymru IP to ASN lookup v1.0 (2019).
URL `http://asn.cymru.com/`

[27] Y. Tian, R. Dey, Y. Liu, K. W. Ross, Topology mapping and geolocating for China's Internet, IEEE Trans. on Parallel and Distributed Systems 24 (9) (2013) 1908–1917.

[28] H. Parmar, M. Thornburgh, Adobe's real time messaging protocol, `http://www.adobe.com/devnet/rtmp.html` (Dec. 2012).

[29] Adobe, Adobe flash video file format specification versions10.1, `http://www.adobe.com/devnet/f4v.html` (Sep. 2010).

[30] Z. Zhu, W. Lu, L. Zhang, N. Ansari, Dynamic service provisioning in elastic optical networks with hybrid single-/multi-path routing, J. of Lightwave Technology 31 (1) (2013) 15–22.

[31] L. Gong, Z. Zhu, Virtual optical network embedding (VONE) over elastic optical networks, J. of Lightwave Technology 32 (3) (2014) 450–460.

[32] Douyu livecast danmu dataset (2018).
URL https://doi.org/10.7910/DVN/X5SCPM

[33] Douyu, Douyu third-party developper API document v2.1 (Apr. 2016).
URL http://dev-bbs.douyutv.com/forum.php?mod=viewthread&tid=113

[34] C. Spearman, The proof and measurement of association between two things, American J. of Psychology 15 (1904) 72–101.

[35] F. Dobrian, A. Awan, D. Joseph, A. Ganjam, J. Zhan, V. Sekar, I. Stoica, H. Zhang, Understanding the impact of video quality on user engagement, in: Proc. of ACM SIGCOMM'11, Toronto, Canada, 2011, pp. 362–373.

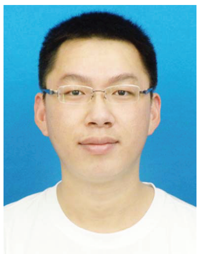
[36] H. He, Y. Zhao, J. Wu, Y. Tian, Cost-aware capacity provisioning for Internet video streaming CDNs, The Computer Journal 58 (12) (2015) 3255–3270.

[37] M. Gilli, D. Maringer, E. Schumann, Numerical Methods and Optimization in Finance, Elsevier, 2011.

Yuhan Hu received the bachelor's degree in computer science from Anhui University in 2018. She is currently pursuing the master's degree in the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China. Her research interest is focused on measurement and analysis of Internet video streaming systems.

Ye Tian received the bachelor's degree in electronic engineering and the master's degree in computer science from the University of Science and Technology of China (USTC), Hefei, China, in 2001 and 2004, respectively, and the Ph.D. degree from the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, China, in 2007. He joined USTC in 2008 and is currently an Associate Professor with the School of Computer Science and Technology, USTC. His research interests include multimedia networks, Internet measurement, and software-defined networking. He has published over 60 papers and co-authored a research monograph published by Springer. He is the winner of the Wilkes Best Paper Award of Oxford The Computer Journal in 2016. He is currently serving as an associate editor of the Springer Frontiers of Computer Science Journal.

Wen Yang received the bachelor's degree in computer science from Liaoning University in 2017. He is currently a master candidate in the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China. His research work is focused on Internet video streaming.



Xiaodong Wang received the bachelor's degree in computer science from the Nanjing University of Aeronautics and Astronautics in 2015. He is currently pursuing the PhD degree in the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China. His research interests include Internet measurement and the SDN-based technology for the future Internet.

Xinming Zhang received the BE and ME degrees in electrical engineering from the China University of Mining and Technology, Xuzhou, China, in 1985 and 1988, respectively, and the PhD degree in computer science and technology from the University of Science and Technology of China, Hefei, China, in 2001. Since 2002, he has been with the faculty of the University of Science and Technology of China, where he is currently a Full Professor with the School of Computer Science and Technology. He is also with the National Mobile Communications Research Laboratory, Southeast University, Nanjing, China. From September 2005 to August 2006, he was a visiting professor with the Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology, Daejeon, Korea. His research interest includes wireless networks. He has published more than 80 papers in wireless ad hoc and sensor networks.