IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 24, 2017, accepted June 13, 2017, date of publication July 7, 2017, date of current version July 24, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2720189

Crowdsourced Multi-View Live Video Streaming using Cloud Computing

KASHIF BILAL[1,2], (Member, IEEE), AIMAN ERBAD[1], (Member, IEEE), AND MOHAMED HEFEEDA[3], (Senior Member, IEEE)

[1]Department of Computer Science and Engineering, Qatar University, Doha 2713, Qatar
[2]COMSATS Institute of Information Technology, Islamabad 22010, Pakistan
[3]School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada

Corresponding author: Kashif Bilal (kashif@qu.edu.qa)

This work was supported by NPRP through the Qatar National Research Fund (a member of Qatar Foundation) under Grant 8-519-1-108.

ABSTRACT Advances and commoditization of media generation devices enable capturing and sharing of any special event by multiple attendees. We propose a novel system to collect individual video streams (views) captured for the same event by multiple attendees, and combine them into multi-view videos, where viewers can watch the event from various angles, taking crowdsourced media streaming to a new immersive level. The proposed system is called Cloud-based Multi-View Crowdsourced Streaming (CMVCS), and it delivers multiple views of an event to viewers at the best possible video representation based on each viewer's available bandwidth. The CMVCS is a complex system having many research challenges. In this paper, we focus on resource allocation of the CMVCS system. The objective of the study is to maximize the overall viewer satisfaction by allocating available resources to transcode views in an optimal set of representations, subject to computational and bandwidth constraints. We choose the video representation set to maximize QoE using Mixed Integer Programming. Moreover, we propose a Fairness-Based Representation Selection (FBRS) heuristic algorithm to solve the resource allocation problem efficiently. We compare our results with optimal and Top-N strategies. The simulation results demonstrate that FBRS generates near optimal results and outperforms the state-of-the-art Top-N policy, which is used by a large-scale system (Twitch).

INDEX TERMS Cloud, crowdsourcing, multi-view video, QoE, resource allocation.

I. INTRODUCTION

Advances in rich media generation devices and wireless networks have led to the massive increase in crowdsourced media generation [1]. Portable video capturing devices, such as cellular phones are commonly used to capture and upload various events for viewers across the globe in real-time. Live video streaming supported by various popular media websites, such as YouTube Live and Twitch offers a convenient platform to broadcast live streams to a substantial number of viewers around the world [2]. From 2015 onwards, mobile crowdsourced live streaming received a massive increase with the launch of various mobile applications, such as Meerkat, Periscope, and YouNow [3]. Using such applications, an attendee can capture an event (e.g., game or concert), and makes it available to remote viewers in real-time. For example, when Meerkat was initially integrated by Twitter to broadcast live videos to followers and friends, 28,000 users used the service in the very first week [4]. Another example is Periscope which is named by Apple as application of the year 2015. Around 100 Million live broadcasts were hosted by Periscope in three months from January to March 2016 [5].

In conventional single-view video, the viewer gets one view of the video from just one angle without any possibility to watch the scene from other angles. Multi-view videos on the other hand, are composed of multiple video streams captured simultaneously using multiple cameras from various angles (different viewpoints) of a scene [6]. Multi-view videos offer more appealing and realistic view of the scene leading to higher user satisfaction and enjoyment. However, displaying realistic and live multiview scenes captured from a limited view-points faces multiple challenges, including excessive number of precise synchronization of many cameras, color differences among cameras, large bandwidth, computation, and storage requirements, and complex encoding [21]–[23]. Most current multi-view video setups are very limited and based in studios. We propose to exploit crowd-sourced videos to offer multi-view video stream. Multiple individual video streams captured by crowdsourcers

watching an event can be aggregated to provide a multi-view experience for remote viewers. Live crowdsourced multi-view streaming enables viewers to watch the event from multiple angles in real-time. This multi-view streaming will bring greater immersion, more feeling of being physically there, and will allow experiencing events as never before.

In the proposed system called Cloud based Multi-View Crowdsourced Streaming (CMVCS), multiple contributors watching an event (e.g., game, concert) capture the event from different angles and broadcast their captured streams to the system in cloud. The CMVCS then handles the user interaction and delivers the required views to viewers based on their view navigation and requested viewpoint. Compared to traditional single view crowd-soured streaming systems [2]–[5], CMVCS is more complex, as it is composed of multiple streams coming from different contributors capturing an event, which need to be processed, transcoded, and delivered to viewers watching different views of the same event. Each view stream needs to be transcoded to various representations, so that viewers with different devices and bandwidth capabilities can enjoy the views at higher user satisfaction and Quality of Experience (QoE). CMVCS brings multiple new challenges that need to be tackled in real-time in an efficient manner, such as view acquisition, selection of potential views for transmission, resource allocation, and respective video quality representations to maximize QoE.

Various video content providers use a specific set of representations to transcode the video stream in various qualities. Viewers with higher network capacity can be served with higher resolution and bitrate representations, and vice versa. A study [7] on large scale crowdsourced live streaming system (Twitch.tv) shows that Twitch receives crowdsourced videos from more than 100 countries spanning over 150 different resolutions. Moreover, the number of live streams and viewers watching the video streams fluctuate widely over time [7], so elastic and pay per use characteristics of cloud computing offer a viable solution for resource provisioning with varying requirements. To achieve the highest level of QoE, each view should be encoded to all representations. However, in such a case, Twitch having on average around 10,000 channels online will require 50,000 cloud instances (each transcoding a single video channel) to transcode each video in all five representations, which is infeasible. Therefore, the views and set of representations should be selected based on the popularity of the view and the available resources. For instance, Twitch uses Top-N strategy to transcode the premium users' videos (around top 300) to all possible representations [7], whereas the remaining streams are transcoded to one representation (source) only.

We define the resource allocation problem addressed in this paper as: *to choose a set of video streams with each video transcoded to a set of representations to maximize the user satisfaction considering the computational and communication resource constraints*. We formulate and solve this resource allocation problem using Mixed Integer Programming (MIP) based multi-constrained, multiple choice Knapsack strategy to find the optimal resource allocation of views to representations. However, considering the size of the problem with tens of thousands of streams and millions of users, the optimal solution is very expensive. Therefore, we propose a heuristic algorithm called Fairness Based Representation Selection (FBRS) to efficiently choose the set of views and their representations based on the average popularity of views. The available resources are divided fairly among the streams based on the popularity share. We compare the optimal, Top-N, and FBRS resource allocation algorithms considering various computational and communication limits to assess the performance of the FBRS solution in various scenarios. We use real-world video traces collected from Twitch.tv in 2015. The results are analyzed based on the overall QoE score. Previous video QoE metrics for streaming videos [7], [9] did not consider the viewer's network capability. Therefore, we developed a QoE metric specifically focusing on the video representation received by viewer and bandwidth capacity of the viewer. To achieve the highest QoE value, i.e., 1, the viewer should be provided with the representation that best matches her bandwidth capability. If the viewer receives a representation that is less than her bandwidth, then the QoE is decreased. Our simulation illustrated that FBRS produces near optimal results in real-time and outperforms Top-N strategy in various scenarios.

The main contributions in this paper are summarized as:

- We present the design and architecture of a Cloud based Multi-View Crowdsourced Streaming (CMVCS) system that allows viewers to experience the captured events from various angles.
- We propose a QoE metric to determine the overall user satisfaction based on the received view representation and the viewers' bandwidth capability.
- We formulate a Mixed Integer Programming (MIP) optimization problem for resource allocation to choose the optimal set of views and representations to maximize QoE in constrained settings.
- We propose a fairness based heuristic algorithm to find near optimal resource allocation efficiently.
- We use multiple real-world traces to simulate various scenarios and show the efficiency of the proposed solution.

The rest of the paper is organized as follows. CMVCS design and its major modules are discussed in Section II. Our proposed QoE metric and resource allocation problem formulation is detailed in Section III. Section IV presents the proposed solutions using MIP and FBRS heuristic. The trace-based evaluation is presented in Section V, and Section VI concludes the paper.

II. CMVCS DESIGN

Crowdsourced live streaming is gaining popularity and millions of viewers' watch the live video streams generated by novice users. Various crowdsourced platforms are already being used heavily for live streaming, such as Twitch, YouTube Live, Meerkat, Periscope, and YouNow. However,

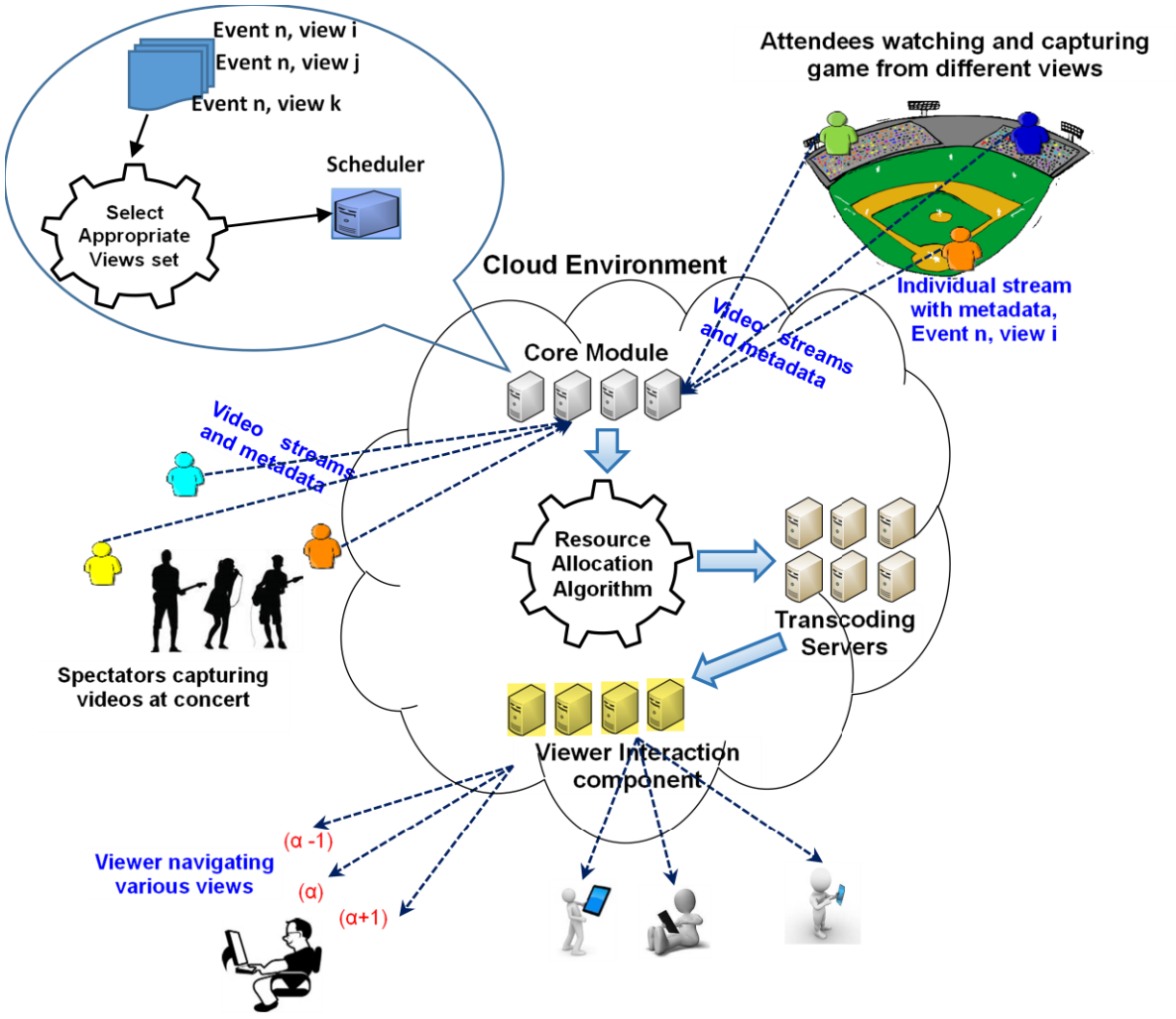

FIGURE 1. Crowdsourced multi-view live streaming syste.

none of these popular platforms offers crowdsourced multi-view video streaming. Considering the proliferation of media capturing devices, network advancements, and elastic cloud capabilities, a crowdsourced multi-view live streaming platform is feasible and practical. Existing crowdsourced streaming systems and applications may also be enhanced by adding the multi-view streaming features. In this section, we provide generic architecture of CMVCS system.

The CMVCS system is comprised of three major modules (See Fig. 1): (a) views capturing module (crowdsourcers), (b) core module, and (c) scheduler. Multiple crowdsourcers (also referred as users) capturing a common event constitute view capturing module. Existing live streaming platforms, like Meerkat and Periscope already have their applications available for popular platforms, such as Apple and Android. However, such applications are designed for single user, for streaming personal videos, without any interaction with other users. Therefore, multi-view CMVCS needs an application to support multiple viewers and required metadata to support multiple views. The application should facilitate users to capture the video and upload the video along with related metadata, such as location (Global Positioning System (GPS) coordinates), time, capturing angle, name of the location and captured event along with any other useful information set by the application to help organize the captured video in a multi-view event. Today smart phones are equipped with various sensors, such as gyroscope, accelerometer, compass, and GPS. The application uses information from these sensors to accurately estimate the position and capturing angle of the device, along with any movements or change in angle or position. The video stream and captured data are uploaded to a server in the core module.

The core module is responsible for: (a) interacting with crowdsourcers, (b) receiving captured video streams and metadata from crowdsourcers, (c) organizing multiple views into events, (d) viewer interaction and sending transcoded views to viewers. Core module acts as a user facing module, which interacts with crowdsourcers and receives user captured videos and metadata. When a user starts capturing an event, the users' GPS location is sent to the core module.

The core module looks for the available events in the close vicinity of new user based on users' location and sends the list of available events to user to choose an event. If the user is capturing an available event, the user makes the selection and starts capturing the event. If the user is capturing a new event, the user provides the event title and a summary of the event and then submits it to the core module. Upon receiving a new video stream, the core module analyzes the accompanying metadata of the stream, and extracts the crowdsourcer's GPS location and capturing angle. The core module matches the received GPS location with the selected event location to ensure the correct association of the received video stream to an event. In case of the lack of appropriate labeling of the event, the core module can also organize multiple views of an event. For instance, received location of multiple videos can be matched to check if the video lies within a limited radius, (e.g., 100m) to assume the views belong to a single event. A simple example can be a football match where various attendees may be capturing the live game. All users should lie within a limited radius, therefore, the core module can identify the event and confirms that all users are capturing the same event. Besides organizing multiple views in a single event, the core module should also choose the best quality video captured at the most appropriate angles to ensure the highest quality and coverage by analyzing the metadata periodically. The videos capturing same views can be identified by the capturing angles. Moreover, the number of views and the angle distance between two views can be defined adaptively based on the number of available streams for an event. After choosing all streams captured for a specific event, the core module organizes the video streams based on the captured angle of the video, where each distinct capturing angle will serve a specific view for viewers (called views onwards). The selected views are sent to the scheduler module. The next step is to transcode the captured views in various representations for Adaptive Bitrate Streaming (ABR), such as DASH streaming server to serve the viewers at the best matching representation and bitrate. For instance, Twitch TV uses five representations [7]. However, considering the limitations on transcoding servers and system bandwidth, not all of the representations are possible. Moreover, not all views are being watched by viewers. Potential views need to be chosen carefully to find the representation settings to maximize the viewer's QoE and minimize the number of required transcoding servers and bandwidth.

The core module sends the updated popularity of views (number of viewers watching a specific view) to the scheduler, which periodically selects the views to be transcoded, and the representation set for the considered views based on the popularity of different views. Popular views would be transcoded to multiple representations, so that most viewers can get the matching representation according to their bandwidth limits. Views with low popularity are transcoded in the least quality representation to serve all users and to adhere to the resource constraints and maximize overall QoE by allocating more resources to popular views. Any view with no viewer will not be transcoded to save resources.

Multi-view videos are generally transcoded using Multi View Coding (MVC) [19] or simulcast [20]. In MVC, all captured views are encoded as a single video, which offers view switching. In simulcast, each view is encoded separately and independently for transmission to viewers. MVC based encoding is not feasible in CMVSC because of heterogeneity in devices and captured view quality, lack of synchronization and calibration, and lack of consistency in capturing time and angle of the views. Moreover, MVC requires special encoders and decoders, which encodes all of the views together, therefore, the overall size of the video is larger than a single simulcasted view. Furthermore, MVC encoded video is vulnerable to quality degradation in case of frame loss [21]. On the contrary, simulcast uses standard encoders to encode single view separately, and is smaller in size when considering a single view transmission. Moreover, simulcast is more quality resilient as compared to MVC. Therefore, simulcast based transcoding is more feasible for CMVSC system.

We assume that at a time instance t the viewer may switch to any of the two adjacent views of currently watched view (reference view α), i.e., either to right $(\alpha + 1)$ or left $(\alpha - 1)$. Therefore, all views adjacent to the views currently being watched by viewers will be transcoded with at least minimum bitrate representation even if no viewer is viewing them to provide swift view delivery. View switching request is received by view delivery and viewers' interaction component of core module. Adjacent views of all watched views are also transcoded, therefore, view delivery module starts sending the requested view and updates the statistics of view popularity to the core module, which in turn updates the information to scheduler for optimizing view transcoding. As transcoding is a computational intensive problem [7], therefore, we allocate a single cloud instance for transcoding a view in one of the representations (cloud instance and transcoding server are used interchangeably).

The CMVCS is a complex system, with each module having its respective challenges. For instance, in view capturing module, crowdsourcers depict massive heterogeneity in terms of devices, capturing quality, and available bandwidth to upload the video, etc. Moreover, the crowdsourcers may stop capturing the view at any time, or change the capturing angle. Multiple contributors may be capturing the event from similar angle, uploading duplicate views to the system. The core module has to identify and select best views considering various angles to deliver the highest possible coverage and video quality. Moreover, duplicate streams need to be identified and the best stream among them needs to be sent to scheduler. Furthermore, core module has to constantly check the change in capturing angle of the crowdsourcers, identify backup streams for transmission in case the chosen crowdsourcers changes the capturing angle or stops capturing or uploading the video. The core module has to consider all these issues to choose the best views for delivery. As the received views from capturing module, popularity of views, and viewers'

available bandwidth are expected to fluctuate, therefore, the scheduler needs to periodically update the chosen views and their respective representation set. Even if we assume that the set of captured views are constant, popularity of views is expected to change considerably by viewers switching various views within an event, as discussed in [24].

In this section, we presented a generic CMVCS system architecture highlighting its various modules and tasks. In this study we focus on the scheduler module and resource allocation problem. To achieve maximum user satisfaction, we formulate the system as a multi-constrained optimization problem. We solve the resource allocation to compute the best representation set for each view by using multi-constrained multiple choice knapsack algorithm. The problem is multi-constrained as we consider both computational and bandwidth limits while allocating resources for view transcoding. The problem is multiple choice knapsack problem, because, system needs to transcode each considered view to at least one representation. However, popular views are transcoded to multiple representations. The detailed problem formulation is discussed in the following section.

III. PROBLEM FORMULATION

In a set of $\mathbb{E} = \{e_1, e_2, e_3, \ldots, e_n$ events, multiple crowdsourcers capture and upload different views of the same event. Set of views captured by various crowdsourcers for an event e_i are denoted by set $\mathbb{V} = \{v_1, v_2, v_3, \ldots, v_m$. Each stream v_i is uploaded to a core module server f_i within cloud. A set of metadata variables $\mathbb{M} = \{m_1, m_2, \ldots, m_k$ representing information related to the uploaded view is also sent periodically to the server. Metadata information may contain physical location, event id, sourcer id, view id, capturing view angle, video quality, data rate, etc. The core module sends this metadata to scheduler to select the distinct potential views and assign cloud instances to process/transcode the selected views based on the popularity of view.

Let $\mathbb{U} = \{u_1, u_2, u_3, \ldots, u_j\}$ represent the set of viewers watching the events. $\mathbb{R} = \{R_1, R_2, \ldots, R_n$ represents the transcoding representation set in Kbps, e.g., $\mathbb{R}$ includes 360, 480, 720, 1080, and 4K, which are possible video representations (as in case of Twitch) [7]. In such as case, five representations of view v_i will deliver maximum satisfaction level. However, based on the viewer's distribution and resource availability/cost constraints, the selected set of representations may be less than five. Let $\mathbb{r}_i (\mathbb{r}_i \subseteq \mathbb{R})$ be the set of representations for view v_j in an evente_k. The adjacent views, $v_{\alpha+1}$ and $v_{\alpha-1}$(v_α is the view being watched by a viewer) are decided based on the viewers switching behavior and v_α. A change in v_α may also mandate a change in r_i, considering various viewers watching v_α. Therefore, this interactive multi-view live video streaming system is highly dynamic and adaptive as compared to legacy live streaming systems, and requires periodic updates in resource allocation.

Various QoE metrics have been proposed to quantify the viewers' QoE [7], [9], [18]. However, most of the proposed QoE metrics do not consider the available bandwidth of the viewer [18]. In case of live streaming, viewer's available bandwidth is vital, as viewer can only receive a video representation respective to available bandwidth. He *et al.* presented a viewer satisfaction metric based on number of representations. The authors considered maximum 5 representations for a live streaming system and derived average satisfaction value for viewers. The authors considered that if the system offers all 5 representations (that is maximum number of representations), then every viewer will receive a representation matching one's bandwidth. However, He *et al.* did not consider the bandwidth or network characteristics of the viewers watching a specific view, and calculated overall satisfaction score for all of the viewers collectively based on the number of representations. In a scenario, where most of the viewers either have very high bandwidth availability, e.g., Korea, where approximately more than 78% of the viewer have more than 10Mbps Internet connectivity and average bandwidth of the users is 26Mbps [12]. In such areas, only a single or two representations might be able to satisfy >80% of the viewers gaining higher overall satisfaction (without providing all 5 representations). Similarly, in areas like India, where 70% of the users have 4Mbps or less Internet connectivity, couple of low bitrate representations will be able to satisfy >70% of the viewers. Therefore, calculating average viewers' satisfaction independent of their bandwidth may lead to wrong QoE estimations. We extended the QoE metric presented in [7] for individual viewer considering the viewer's bandwidth and the video representation she receives. Logarithmic shaped function of our proposed metric closely matches the ones presented in [7] and [9].

Let $\mathcal{S}_{(\mathbb{r}_i)}$ be the satisfaction level of the view representation based on quality being received and maximum quality receivable by a viewer. We calculate the user satisfaction level based on the extent of viewer's receivable quality and received quality of video. For instance, if a viewer is receiving a view v_j in 720p representation from server, whereas the viewer has enough bandwidth to receive 1080p or 4K stream, then the satisfaction level is less when fewer or low quality representations are produced. This would be the case when the system transcodes the view v_j in lower representation formats because of low popularity of the view and does not have higher quality representations. Let $l = |\mathbb{R}|$ represent the total number of possible video representation set, ($l \leq 5$ in this case), $a = \frac{1}{l}$, represent extent of loss in video quality, l_r is the quality level (bitrate representation) received by viewer, (i.e., 360, 480, ..., 4K), and l_u is viewers' maximum receivable representation level, based on viewers available bandwidth. Then, the loss in video quality is:

$$d_u = |l_u - l_r| \times a. \tag{1}$$

The receivable video quality for the user can be then formulated as:

$$\mathrm{q}_u = \mathrm{q}_{max} - d_u, \tag{2}$$

where q_{max} is the maximum receivable quality of video set to 1. If there is no loss, i.e., when the user receives the same

TABLE 1. QoE based on user satisfaction values.

		Received Video representation				
		4K	1080	720	480	360
User Bandwidth Capability	4k	1	0.9	0.77	0.6	0.301
	1080	-	1	0.9	0.77	0.6
	720	-	-	1	0.9	0.77
	480	-	-	-	1	0.9
	360	-	-	-	-	1

TABLE 2. Notations used in formulation.

Name	Description	Name	Description
$\mathbb{E}$	Set of events	$\mathcal{S}$	Satisfaction level for representation
$\mathbb{V}$	Set of views	$\mathbb{B}$	Total bandwidth capacity
$\mathbb{U}$	Set of users	$\mathbb{C}$	Total computational instances in cloud
$\mathbb{R}$	Representation set	b_r	Received representation bitrate
$\mathbb{M}$	Set of metadata	b_u	Viewers' bitrate

representation quality as her available bandwidth capability (when $d_u = 0$), then q_u will be 1.

The satisfaction level of user can then be formulated based on satisfaction and received utility in [7] and [11] as:

$$\mathcal{S} = U_{max} - |log(q_u)|\,, \tag{3}$$

where U_{max} represents the highest satisfaction level, set at 1. The satisfaction function for the users with various network bandwidth capabilities, and received video quality representations can be seen in Table 1. The notations used in the problem formulation are presented in Table 2. Based on the user satisfaction level presented in Eq. (3), the view representation set allocation problem can be formulated as:

$$\text{Maximize} \quad \sum_{u\in\mathbb{U}}\sum_{e\in\mathbb{E}}\sum_{v\in\mathbb{V}}\sum_{r\in\mathbb{R}} S_{uevr} \times I_{[uevr]}, \tag{4}$$

Subject to :

$$b_r \leq b_u \tag{4a}$$

$$b_u \geq b_0; b_0 = 360p \tag{4b}$$

$$\sum_{u\in\mathbb{U}}\sum_{e\in\mathbb{E}}\sum_{v\in\mathbb{V}}\sum_{r\in\mathbb{R}} b_r \times I_{[uevr]} \leq \mathbb{B} \tag{4c}$$

$$\sum_{e\in\mathbb{E}}\sum_{v\in\mathbb{V}}\sum_{r\in\mathbb{R}} b_r \times X_{[evr]} \leq \mathbb{C} \tag{4d}$$

$$\sum_{v\in\mathbb{V}}\sum_{r\in\mathbb{R}} X_{[evr]} \geq 1 \tag{4e}$$

$$I_{[uevr]} \in [0, 1] \tag{4f}$$

$$X_{[evr]} \in [0, 1] \tag{4g}$$

Indicator function $I_{[uevr]}$ serves as decision variable

$$I_{[uevr]} = \begin{cases} 1, & \textit{if user } u \textit{ receives view } v \textit{ of event } e \textit{ at } r \\ 0, & \textit{otherwise} \end{cases}$$

$$X_{[evr]} = \begin{cases} 1, & \textit{if view } v \textit{ of event } e \textit{ is transcoded with } r \\ 0, & \textit{otherwise} \end{cases}$$

The objective function is to maximize the overall QoE. The QoE is defined as a function of viewer's satisfaction. b_r and b_u represents the received representation's bitrate and viewer's available bandwidth, respectively. Where, B and C represent the total bandwidth capacity and total number of cloud instances. The constraint (4a) puts a limit that the received representation bitrate must be less than or equal to the viewer's bandwidth capacity. Eq. (4b) puts a constraint for considering viewers capable of receiving a minimum considered representation bitrate, i.e., 360p. Constraint (4c) forces a limit on the system bandwidth, i.e., the overall bandwidth utilized for serving all viewers at various representations must be less than or equal to the bandwidth capacity imposed by the server. Each view transcoded in a specific representation requires one cloud instance, constraint (4d) places a limit on total number of transcoded representation in system. In this way, the selected views will be transcoded in multiple bitrate representations and some views in just one representation based on the available cloud instances. Constraint (4e) converts the optimization problem to a multiple choice problem, where optimizer has to select at least one representations for each view.

IV. PROPOSED SOLUTION

A. MULTI-CONSTRAINED MULTIPLE CHOICE KNAPSACK

We implemented the problem in ILOG CPLEX general solver [10] to find the optimal resource allocation based on the computational instance and bandwidth constraints. The aggregate user satisfaction obtained by using a specific set of representations are considered as profit or gain, and consumed number of cloud instances and aggregate bandwidth for all users serve as weights. For instance, if a view is transcoded to all five representations, then the weight will be 5, and viewers' QoE will be maximum, as all of the viewers will get the desired representation. However, if a view is transcoded in just one representation, then the weight will be 1, and aggregate user QoE will be ≤ 1, depending on the bandwidth capabilities of the viewers watching that view.

The profit values serve as the major selection criteria. Considering n number of representation in set R, a single view can be transcoded into 2^{n-1} possible combinations considering requirement to fulfill all of the viewer's bandwidth constraints. For instance, if a chunk of viewers can only receive 360p video, then the selected representations set must have 360p representation. If there is only one computational instance available, then the only possible representation is 360p, as only this representation fulfills constraint (4a) for all of the viewers. Though, the overall QoE for this representation will be less as viewers capable to receive higher

bitrate representations are forced to watch 360p representation. If the view can be transcoded into two representations, then the possible set to choose a representation set is {{360, 480}, {360, 720}, {360, 1080}, or {360, 4K}}. 360p represents the least bitrate representation required to serve the users having capability to receive 360p only. Otherwise, the least representations may be composed of higher bitrates based on viewers. In this way, for $n = 5$, there are maximum 16 possible representation combinations.

B. FAIRNESS BASED REPRESENTATION SELECTION (FBRS)

Considering the interactive nature, size, and real-time requirements of the problem to identify the best set of representations to maximize QoE, knapsack based optimal implementation is infeasible. Therefore, we propose a heuristic algorithm FBRS to calculate near optimal results in real-time. We use the optimal results obtained from CPLEX solver as a benchmark to gauge the FBRS efficiency. FBRS is a fairness based policy, where the views are given priority based on their popularity. In this way, popular views get more resources in terms of computational instances and bandwidth to be transcoded to multiple representations to increase the overall QoE.

Algorithm 1 presents FBRS. The input to FBRS algorithm is set of views along with viewers' information. The algorithm decides which views are transcoded in what set of representations. l represents the total number of representations, n represents total number of views, and k is total number of viewers watching a view v_i. C represents the total number of computation instances, whereas B represents total available bandwidth capacity. The algorithm first ranks all views based on view popularity in decreasing order in line 1. Ranking of views will give higher priority for representation assignment to popular views. As all of the considered views should be transcoded to at least one representation, therefore, algorithm assigns one computational instance and necessary bandwidth to all of the considered views to transcode the view in lowest requested bitrate representation r_{min}. The computational instances and bandwidth used to transcode r_{min} are subtracted from available resources. After allocating one representation to all of the views, the algorithm estimates the popularity share for each view. Popularity share is the ratio of viewers watching view v_i to total number of viewers. Based on the calculated ratio, resources are assigned to each view in line 6. In this way, popular views are eligible for more resources than unpopular ones. In case of bandwidth allocation, one may restrict sharing of resources to very small chunks, e.g., less than 10Mbps, as these small chunks may not be useful for any representation. However, considering the long tail data set, where thousands of views have very little popularity and viewers, such small chunks may be assigned to a popular view in aggregated form. Therefore, such bandwidth is added to extra bandwidth resource pool B_x. The threshold may be based on bandwidth or number of viewers etc. If the allocated computational resources are in excess, then the extra computational instances are added to extra pools of computational resources C_x in line 8. These extra resources are given to those views which have less resource than required for all representation transcoding. Based on available resources, the representation sets are selected prioritizing the representations that are demanded by most of the viewers in line 13-15. Bandwidth constraints are also checked in line 13, and if allocated bandwidth is less than required, then the bandwidth may be borrowed from B_x. The excess bandwidth after transcoding is also added to B_x. FBRS algorithm complexity is $O(n \times C_i)$, where n is total number of views and C_i represents total number of chosen representations based on the popularity share of the view $C_i \leq 4$. As the least bitrate representation is already selected for every view, therefore, at most 4 remaining representations may be considered. Based on the resource availability, on average C_i, is a small value because the view popularity follows a long tail distribution. With a large number of views having very low viewers count. The chosen set of representations are used in Algorithm 2 to calculate the aggregate QoE. The QoE is calculated for each representation in SelectedReps. $z = |k_{r_i}|$ represents the total number of viewers capable of receiving representation r_i. For instance, if $r = \{360, 1080\}$, then viewers with bitrate capacity of 360, 480, and 720 will receive 360p representation meeting the constraint (4a), and viewers with 1080 and 4K capable bitrates will receive 1080p representation. Similarly, the aggregate QoE for all viewers will be calculated based on Eq. (3).

V. TRACE BASED EVALUATION

A. DATA SETS

We used real-world trace workloads to compare the performance of FBRS in various scenarios and viewer distributions. We used trace data captured by the Twitch API captured on Mar. 1, 2015 at 7:30am. The same set of data has been used in [7] for simulation settings. Although, Twitch data does not provide multi-view events. Nonetheless, we used Twitch data traces for two reasons. First, due to unavailability of any multi-view live event streaming data, Twitch data provides at least a realistic count for the number of streamed videos and active viewers. Second, Twitch data is studied and analyzed in detail depicting that such data holds similarity to some of the events where most of viewers watch a small set of videos, such as a football game or a set of significant events held at different places around globe. As the Twitch data is single view, we have converted the single view data to multi-view data by gathering the similarity (based on the played game, e.g., Dota 2, League of Legends, and Dying Light etc.) in videos wherever possible. Each event is assigned 12 views, among which user can navigate.

Besides Twitch workload traces, we also simulated three data sets used by Mukerjee *et al.* [8]. Trace is collected from a service provider for one-hour video records. First dataset depicts an average day video usage summary for one-hour trace from a service provider. The trace consists of 4,144 videos, requested by 18,837 users globally from

TABLE 3. Viewer bandwidth probability distribution and bitrates.

Representation	360p	480p	720p	1080p	4K
Probability	0.1	0.17	0.18	0.34	0.21
Bitrate (Mbps)	0.4 – 1	0.5-2	1.5 - 6	3 - 9	>15

2,587 cities. Top 7% of the videos in the data set were watched by 50% of the viewers. Second data set is a long tailed workload comprised of 10,000 videos requested by 82,000 clients. The workload depicts that 99% of the long tailed workload videos are requested by 60% of viewers. Whereas 1% of videos are requested by 40% of the viewers. Third data set is a heavy head distribution workload exemplifying some significant events, such as sports events that are watched by massive number of viewers. The heavy head workload holds same count of 10,000 videos, but watched by a very huge viewers base of 4 Million viewers worldwide. Considering a very small chuck of views, i.e., 1% is watched by 99% of the viewers, the workload illustrates a heavy head distribution [8]. A similar 12 views per event ratio is used in these data sets as well.

We simulated these workload traces with their respective video and viewers' settings. As the user's bandwidth capacity is a pivotal part in our system, we assign each viewer a network capacity based on the latest 2016's first quarter state of the Internet report [12]. The reports reveals that 73% of the users across the globe have 4Mbps or higher Internet connectivity. Similarly, 38% of users have 4Mbps - 9Mbps connectivity, 35% holds a connection of 10Mbps or higher with 21% having 15Mbps or higher, and 8.5% having 25Mbps or higher Internet connectivity. Based on the figures available in report, we assign the probability for bandwidth capability of a viewer to watch a specific representation as shown in Table 3.

Bitrates for the representations are taken from YouTube Live encoder settings [13]. As the Internet connection are generally available in packages of 1Mbps, 2Mbps, 4Mbps, 8Mbps, 10Mbps etc. Therefore, for 27% of users having connection below 4Mbps, we divide the viewers in two groups having 0.1 and 0.17 probability for 1Mbps and 2Mbps Internet connectivity, respectively. Similarly, all of the remaining probabilities are assigned based on the state of the Internet report and YouTube Live encoder settings.

B. RESULTS

We performed detailed simulations to compute the representation set selection and overall QoE based on FBRS allocation scheme. The obtained results are compared with the optimal representation selection (QoE generated by CPLEX optimizer) and Top-N policy. Top-N is the strategy used by Twitch to transcode top 'n' videos (from premium) users to transcode video in all available representations [7]. As Twitch premium members have a requirement to have 500 regular viewers.

Algorithm 1 Fairness Based Representation Selection

Input: Views of all events and viewer information
Output: Representation allocation for each view
$l = |\mathbb{R}|$
$n = |V|$
$k = |u_{v_i}|$
1. Rank all views in decreasing order based on popularity
2. Allocate one computational instance to transcode r_{min} for all views
3. $C^{'} = C - |C_0|$
   $B^{'} = B - |B_0|$
4. for i from 1 to n
5. Calculate popularity share p_i for view v_i
   $p_i = |u_{v_i}| / \sum_{j \in V} |u_{v_j}|$
6. Calculate resources share for view v_i
   $C_i = p_i \times C^{'}$
   $B_i = p_i \times B^{'}$
7. If $(B_i < I)$
   $B_x = B_x + B_i$
   $B_i = 0$
   End If
8. If $(C_i \geq l)$
   Add extra instances to pool C_x
   $z = C_i - l$
   $C_x = C_x + z$
9. Else if ($C_i < l$)
   Allocate resources from extra resources pool
10. End if
11. Initialize empty SelectedReps queue
12. For j=1 to C_i
13. Get representation set r_j matching highest number of viewers' network capacity and B_i
14. Add r_j to SelectedReps queue
15. End for
17. End for

Most of the popular videos belong to premium members. We present the result for two cases: (a) limited computational instances, where we limit the number of total representation transcoding based on available computational units, without limiting bandwidth, and (b) limited computational instances and limited bandwidth, which is a more realistic scenario as bandwidth is a costly and limited resource.

We start our results analysis with heavy head workload scenario, where viewers around the globe are watching some sports event, such as a football or any other popular event like music concert etc. Heavy head trace contains a video count of 10,000 stream, but with a very large viewer base of 4 Million viewers globally. Only 1% of the views are watched by 99% viewers. We only show the results for heavy head scenario for bandwidth limits, because for unlimited bandwidth, each of the strategy gain QoE value of 0.99 for

Algorithm 2 CalculateQoE

Input: Selected Representation set for view v_i
Output: Aggregate QoE score for v_i
$s = |SelectedReps|$
$z = |k_{r_i}|$
1. $AggQoE_i = 0$
2. For $x = 1$ to s
3. For $y = 1$ to z
4. $d = |y - x| \times a$
5. $q = q_{max} - d$
6. $t = U_{max} - |log(q)|$
7. Add t to $AggQoE_i$
8. End For
9. End For

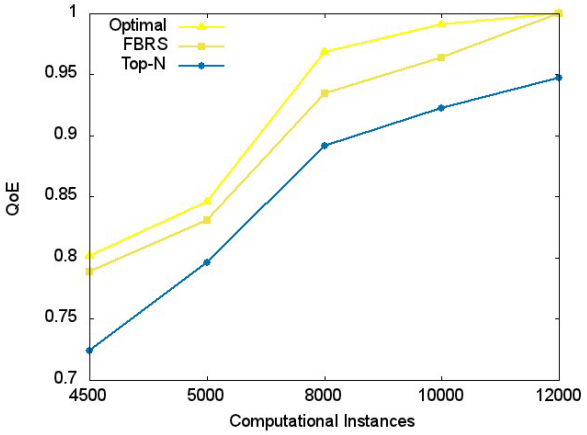

FIGURE 3. QoE based on varying computational instances for Regular data.

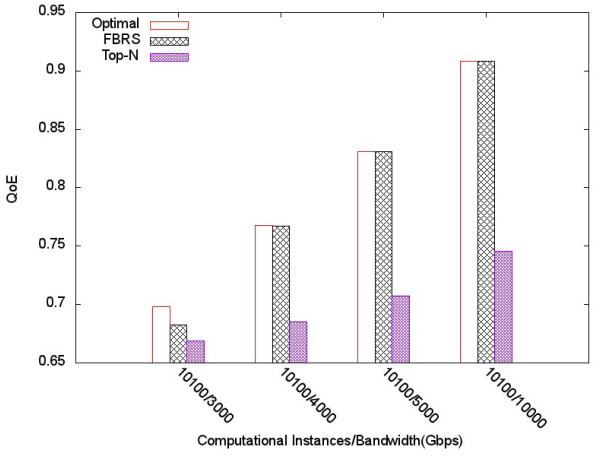

FIGURE 2. QoE based on 10000 computational instances and varying bandwidth for heavy head dataQoE based on 10000 computational instances and varying bandwidth for heavy head data.

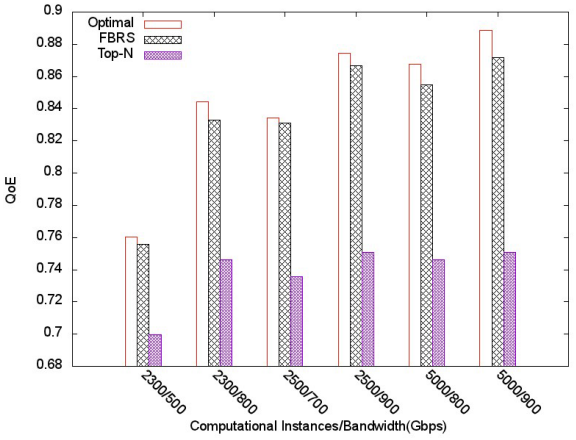

FIGURE 4. QoE based on varying computational instances and bandwidths for regular data.

even 10,500 computational instances, as 99% of the viewers are watching top 1% views. However, for bandwidth limits of 3,000, 4,000, 5,000, and 10,000Gbps, respectively, the FBRS demonstrate prominent lead over Top-N strategy. In such a heavy head scenario with a limited bandwidth (3Tbps – 10Tbps), FBRS gains almost same QoE values as MIP based optimal solution illustrating the effectiveness of FBRS. Both optimal and FBRS gains QoE 0.92 with 10100 computational instances and 10Tbps bandwidth, whereas Top-N is able to achieve QoE value 0.73.

Fig. 3 presents the results for an average day video trace scenario with 18,800 viewers and 4,144 videos from 2,587 cities around the globe. We chose this trace to show a normal day video traffic behavior, where viewers are spread around globe and watching various events. Fig. 3 illustrates the scenario with computational instance limit only, without any restriction of bandwidth. At least 4,144 computational instances are required to transcode the views in a single representation. As the number of computational instances are increased, chosen views are transcoded to multiple representations, increasing the QoE. It can be seen that FBRS outperforms Top-N strategy for both small and large number of computational instances, and attains QoE values close to optimal solution. We also perform simulations with various configurations of computational instances and bandwidth limits. Fig. 4 presents a consolidated result for QoE values for different configurations by imposing varying computational instances and bandwidth limits. We vary computation instances from 4,500—10,000 and bandwidth capacity from 20Gbps—100Gbps. The average bitrate per second estimated in the original trace was 2,725 Kbps. In our simulations, for a bandwidth limit of 50Gbps with 8,000 computational instances, FBRS gains a QoE value of around 0.90 for an average bitrate of 2,783 Kbps. Whereas for the same configuration, optimal and Top-N strategies gain QoE values of 0.93 and 0.81, respectively.

Fig. 5 presents the results for a long or heavy tail video trace generated by Mukerjee *et al.* with 10,000 videos and 82,000 viewers. The popular 1% videos account for 40% of overall requests, whereas rest of the 99% of videos account for 60% of views. Fig. 6 illustrates the results with varying bandwidth and computational instances limits. As can

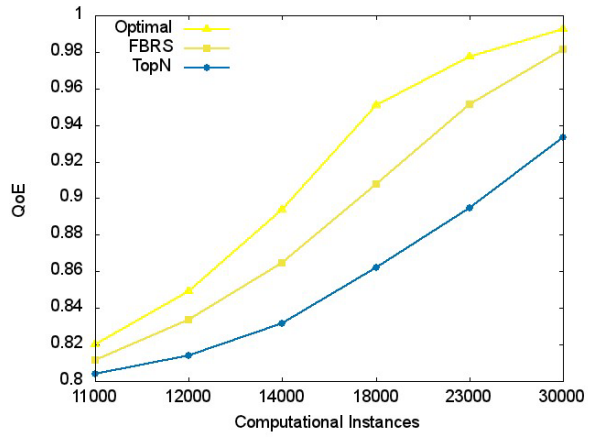

FIGURE 5. QoE based on computational instances for Long Tail data.

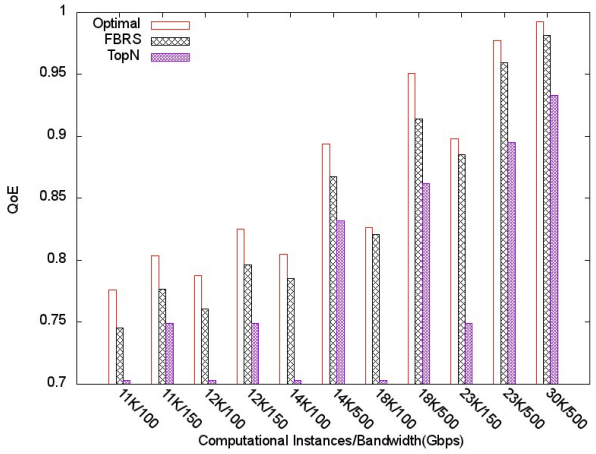

FIGURE 6. QoE based on varying computational instances and bandwidth limits for Long Tail data.

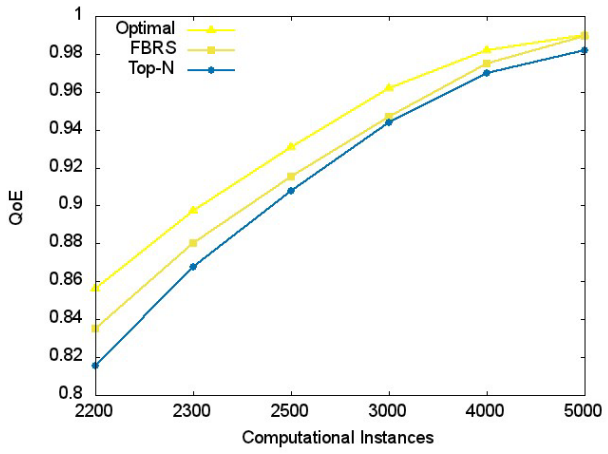

FIGURE 7. QoE based on computational instances for Twitch data.

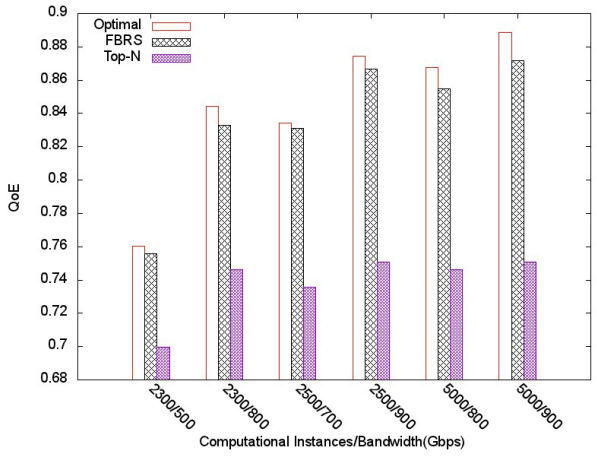

FIGURE 8. QoE based on varying computational instances and bandwidth limits for Twitch data.

be observed from Fig. 5 and Fig. 6 that FBRS adheres to gain better QoE and user satisfaction. Fig. 7 shows the results of the overall viewers QoE for Twitch peak time trace data, i.e., Mar. 1, 2015 at 7:30am. There are total 2,124 views in all events watched by 517,100 viewers. For sake of simplicity, we consider only channels having at least 10 viewers. Fig. 7 illustrates the overall QoE computed using three schemes. It can be observed that FBRS produces near optimal results and outperforms Top-N policy. The reason for a marginal difference between FBRS and Top-N in Twitch workload scenario is the fact that in Twitch data, most of the viewers are watching a small set of videos. For instance, in the considered data set of around 0.5 million users, 0.35 million user watch 3.5% top streams (i.e., 74 channels), rest of 0.15 million viewers watch remaining 96.5% (2050 videos). Therefore, optimizing only 3.5% of streams makes huge difference in the overall QoE.

Fig. 8 presents the results for varying computations instances and bandwidth limits from 500Gbps to 900Gbps to serve 517,100 viewers. It can be seen that when bandwidth limits are imposed, FBRS clearly outperforms Top-N. Moreover, QoE gains for FBRS are very close to optimal results. It can be observed from the results in different scenarios that FBRS efficiently chooses near optimal representation set to serve the viewers and outperforms Top-N resource allocation algorithm used by large scale video content providers.

VI. RELATED WORK

Live video streaming is one of the major Internet bandwidth consumer, anticipated to consume more than 50Tbps [8]. Many live video content providers, such as YouTube Live, Twitch, Periscope, and YouNow serves millions of viewers with live video content. YouTube is serving around more than 1 billion users with around 300 hours' length videos uploaded every minutes [7]. User base at Twitch is around 50 million users watching 150 billion minutes of live stream every month [8]. Cloud infrastructure is now used as the prevailing platform for live streaming systems. The increasing viewer base, watched content, and gigantic bandwidth usage clearly calls for an efficient and intelligent resource

allocation and representations selection schemes. Previous works have addressed the cloud resource optimization for live streaming to minimize the overall cost [1], [7], [14], [15]. Chen *et al.* [1] proposed a strategy to rent cloud resources by optimizing cloud sites in various areas for video transcoding. Wang *et al.* [14] optimized cloud cost by choosing the cloud location for transcoding based on the viewers' locality. Huang *et al.* [15] scheduled the video transcoding tasks inside a cluster based on the video properties. He *et al.* [7] addressed the geo-distributed crowdsourced video contents and video delivery by considering various cloud site across the globe. The authors proposed a QoE metric considering the total number of transcoded representations. However, the authors coarsely calculated the aggregate QoE based on the number of representation produced without considering the viewer's bandwidth capability and viewers' distribution. Similarly, Toni *et al.* addressed the problem of choosing optimal set of video representations to maximize the user satisfaction from video content provider's perspective based on Video Quality Metric score [9], [25]. Toni *et al*. focused on the optimization of the set of representation that should be generated by an ingest server. The authors proposed a theoretical framework to derive optimal set of representations. However, the framework is designed specifically for non-live video on demand (VoD) systems to maximize the user satisfaction under network and system constraints. The authors formulated a Mixed Integer Programming based optimal solution. The authors estimated the user satisfaction as Video Quality Metric (VQM). The authors compared their optimal representation set with the existing recommendation of video representation of Apple, Microsoft, and Netflix, and highlighted shortcomings in the existing video representations recommendation set [25]. Crowdsourced live streaming or choice of multiple representation based on overall viewers are not addressed by them as well. In general, resource cost remained the main optimization parameter in most of the works. Kodera *et al*. proposed a multi-view video streaming using multiple mobile cameras [35]. The study aimed to reduce video traffic between mobiles and access point receiving all streams by using packet over hearing and transmission order control. Each camera receives other camera's video frames and encodes its frame with the overheard frames. The order of camera transmission is controlled by an access point. The evaluation results demonstrated significant decrease in traffic volume with little quality degradation.

Multi-view video is an emerging area where multiple calibrated cameras are used to capture a scene to provide multiview video. Various authors presented server and client based multi-view streaming systems. Server based approaches are adopted by [15] and [16], where the virtual view encoding and synthesis is performed by server, which mandate more storage and bandwidth usage. Client side approaches assign these tasks to client instead of server. Hamza and Hefeeda [6] presented multi-view plus depth based video streaming system based on DASH adaptive streaming. Chakareski designed a constrained optimization framework to maximize the average video quality considering the view popularity by sharing the bandwidth between multiple cameras capturing a multiview scene and a central station (access point) receiving all view streams [26]. The bandwidth allocation follows a receiver driven design, where the central station allocates bandwidth for each camera. The authors considered two scenarios where either a set of chosen cameras transmit the video signal in entirety or partially. In both cases the central station aims to maximize the overall quality. The authors designed a view popularity driven bandwidth allocation scheduling considering the spatiotemporal dependency of the captured views and a data recovery mechanism. The authors emphasized on the video plus depth encoding and error concealment strategy. The authors examined the rate distortion efficiency of the transmission policies. Chakareski *et al*. focused on edge-adaptive wavelet multi-view video coding to target 3D videos with texture plus depth maps [30]. Each user based on one's demand may be serviced by encoding the multi-video based on the users' current status and next view prediction. Whereas, in our case of live video, multiple viewers having different bandwidth capacities needs to be served simultaneously, which require to select a finite number of most feasible bitrate representation to maximize the satisfiability.

Liu *et al*. presented multiple-description coding scheme for transmission of 3D free-view video, with error recovery capabilities [27]. In case of loss-prone wireless links used for video delivery, the authors used a joint temporal and inter-view description recovery mechanism. The description is comprised of four separately encoded sub-streams of data. In [28], the authors presented a network compression based framework for interactive multiview video. The video content is delivered via two disjoint networks paths, which are adapted by a proxy server placed at the junction of both paths to minimize the video distortion at client. An optimization framework to schedule the transmission of multiview video to maximize the reconstruction quality of the video, considering the available bandwidth is presented in [29]. A space-time error concealment strategy was developed to reconstruct the missing content for multiple views. The performance of the system was studied using simulation experiments and the results demonstrated considerable gains over referenced methods in terms of rate-distortion efficiency. Cheung *et al*. addressed the bit allocation problem among texture and depth maps in a depth-image based rendering (DIBR) for multiview images [31]. The objective of the study was to minimize the visual distortion of reconstructed views by distributing texture and depth map bits of the selected views. The authors derived a DIBR based cubic distortion model to estimate visual quality of synthesized views. The experimental results demonstrated that the proposed solution outperforms the referenced schemes with at least 80% reduced complexity. Dorea *et al*. presented an optimized bit-rate allocation scheme for each camera used to capture a multiview video for the free-viewpoint television network [32]. The authors proposed an attention-weighted bit-allocation scheme to allocate more

bits to the cameras have higher popularity and viewer base. Simulation results demonstrated the correctness and efficacy of the attention-weighted rate-allocation scheme over the uniform rate-allocation schemes.

Hamza *et al*. discussed multicasting of 3D free-view video over the wireless networks in [33]. The authors considered a 4G network to multicast two scalable view coded view-plus-depth 3D videos with an objective to maximize the quality of rendered virtual views on viewers' device. Moreover, the authors aim to minimize the energy consumption on mobile receivers by efficiently scheduling the transmission of chosen sub streams. Multicast based multi-view videos delivery is also discussed in [34]. The videos considered in this work are also 3D videos recorded in professional and controlled environment with video along with depth format. The author considered a multicast scenario for the dissemination of source stream and the reconstruct the stream at the client using optimal quality. The authors also handled the packet loss by using optimal channel coding protection levels integrated into source encoding.

Most of these multi-view related works mentioned above are based on calibrated camera settings and professional equipment. The multi-views are generated using professionally synchronized and calibrated array of cameras. In contrast, we aim on non-professional, unsynchronized, non-calibrated, and heterogeneous crowdsourced captured views. Moreover, SVC or multi-view encoding requires synchronization and calibration of captured views which is not possible or very difficult in crowd-sourced multi-views currently. Therefore, MVC based encoding is not feasible in our scenario currently, as discussed in Section II. Furthermore, most of the papers consider VoD, and not the live video. In contrast, we present the idea of crowdsourced multi-view live streaming. Despite being a futuristic idea, the multi-view live streaming is very feasible and practical considering today's smart devices and pervasive connectivity.

VII. CONCLUSIONS AND FUTURE WORK

We proposed a multi-view crowdsourced live streaming system to enable viewers to watch an event from multiple angles. The proposed system significantly differs from legacy multi-view video systems in the fact that multiple views are generated from non-professional crowdsourcers instead of professionally calibrated settings and expensive equipment. We presented cloud based architecture of the system for a scalable and cost effective solution. Moreover, we proposed a QoE metric based on the received video quality and the viewer's bandwidth capabilities. We computed optimal video representation set using our QoE metric by employing MIP based Knapsack optimization algorithm. We proposed FBRS strategy to efficiently compute the representation set for live streams, and compared our results with optimal solution and Top-N strategy used by Twitch. We used real-world video traces to perform experiments. Our results illustrated that considering the potential users to decide the video representations greatly enhances the QoE, and FBRS efficiently produces near optimal results. We compared our results in various scenarios including Twitch peak time trace, an average day video trace, long tail workload, and heavy head workload depicting sports events watched by more than 4 million users. In all of the aforementioned scenarios, FBRS outperformed Top-N strategy and produced near or in some cases equal to optimal results, clearly depicting the superiority of FBRS.

CMVCS is a feasible and practical idea presented in this paper. As mentioned in section II, different modules of CMVCS system has multiple challenges. In this work, we presented a basic architecture of the system along with challenges, and focused on scheduling and resource allocation. In our future work, we aim to focus on viewer's view switching dynamics and minimize the view switching delay by using prediction based view delivery.

ACKNOWLEDGMENTS

The findings achieved herein are solely the responsibility of the author[s]. K. Bilal is on leave from the COMSATS Institute of Information Technology, Pakistan.

REFERENCES

[1] F. Chen, C. Zhang, F. Wang, and J. Liu, "Crowdsourced live streaming over the cloud," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Hong Kong, Apr./May 2015, pp. 2524–2532.

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

[3] Alyssa Newcomb. *Periscope, Meerkat, YouNow*, accessed on Jan. 28, 2017. [Online]. Available: http://goo.gl/hiGnlV

[4] *Turning Live Video Into a Big Deal*, accessed on Jul. 28, 2016. [Online]. Available: http://goo.gl/mNscm9

[5] *Periscope*, accessed on Jul. 28, 2016. [Online]. Available: https://medium.com/@periscope/

[6] A. Hamza and M. Hefeeda, "Adaptive streaming of interactive free viewpoint videos to heterogeneous clients," in *Proc. 7th Int. Conf. Multimedia Syst. (MMSys)*, New York, NY, USA, 2016, p. 10.

[7] 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.

[8] 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," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 311–324, 2015.

[9] L. Toni, R. Aparicio-Pardo, G. Simon, A. Blanc, and P. Frossard, "Optimal set of video representations in adaptive streaming," in *Proc. 5th ACM Multimedia Syst. Conf. (MMSys)*, New York, NY, USA, 2014, pp. 271–282.

[10] IBM. *IBM ILOG CPLEX Optimization Studio*, accessed on Jul. 28, 2016. [Online]. Available: http://goo.gl/96TTRL

[11] C. E. Luna and A. K. Katsaggelos, "Maximizing user utility in video streaming applications," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, vol. 3. Salt Lake City, UT, USA, May 2001, pp. 1465–1468.

[12] Akamai. *State of the Internet Report*, accessed on Jul. 28, 2016. [Online]. Available: https://goo.gl/pfkryb

[13] YouTube Live. *Live Encoder Settings, Bitrates and Resolutions*, accessed on Jul. 28, 2016. [Online]. Available: https://goo.gl/04X4Zb

[14] F. Wang, J. Liu, and M. Chen, "CALMS: Cloud-assisted live media streaming for globalized demands with time/region diversities," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, Mar. 2012, pp. 199–207.

[15] Z. Huang, C. Mei, L. E. Li, and T. Woo, "CloudStream: Delivering high-quality streaming videos through a cloud-based SVC proxy," in *Proc. IEEE INFOCOM*, Shanghai, China, Apr. 2011, pp. 201–205.

[16] E. Kurutepe, M. R. Civanlar, and A. M. Tekalp, "Client-driven selective streaming of multiview video for interactive 3DTV," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1558–1565, Nov. 2007.

[17] T. Maugey and P. Frossard, "Interactive multiview video system with low complexity 2D look around at decoder," *IEEE Trans. Multimedia*, vol. 15, no. 5, pp. 1070–1082, Aug. 2013.

[18] K. Bilal and A. Erbad, "Impact of multiple video representations in live streaming: A cost, bandwidth, and QoE analysis," in *Proc. IEEE Int. Conf. Cloud Eng. (IC2E)*, Vancouver, BC, Canada, Apr. 2017, pp. 88–94.

[19] ISO/IEC 14496-10:2008/FDAM 1:2008(E), *Information Technology—Coding of Audio-Visual Objects—Part 10: Advanced Video Coding, Amendment 1: Multiview Video Coding*, 2008.

[20] G. Cheung, A. Ortega, and T. Sakamoto, "Coding structure optimization for interactive multiview streaming in virtual world observation," in *Proc. IEEE 10th Workshop Multimedia Signal Process.*, Cairns, QLD, Australia, Oct. 2008, pp. 450–455.

[21] M. M. Fouad and H. A. Aly, "A modified multiview video streaming system using 3-tier architecture," *Adv. Elect. Electron. Eng.*, vol. 14, no. 2, pp. 196–204, 2016.

[22] S. Lu, T. Mu, and S. Zhang, "A survey on multiview video synthesis and editing," *Tsinghua Sci. Technol.*, vol. 21, no. 6, pp. 678–695, Dec. 2016.

[23] A. Vetro, T. Wiegand, and G. J. Sullivan, "Overview of the stereo and multiview video coding extensions of the H.264/MPEG-4 AVC standard," *Proc. IEEE*, vol. 99, no. 4, pp. 626–642, Apr. 2011.

[24] K. Diab and M. Hefeeda, "MASH: A rate adaptation algorithm for multi-view video streaming over HTTP," in *Proc. 36th Annu. IEEE Int. Conf. Comput. Commun. (IEEE INFOCOM)*, Atlanta, GA, USA, May 2017, pp. 1036–1044.

[25] L. Toni, R. Aparicio-Pardo, K. Pires, G. Simon, A. Blanc, and P. Frossard, "Optimal selection of adaptive streaming representations," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 11, no. 2s, p. 43, 2015.

[26] J. Chakareski, "Uplink scheduling of visual sensors: When view popularity matters," *IEEE Trans. Commun.*, vol. 63, no. 2, pp. 510–519, Feb. 2015.

[27] Z. Liu, G. Cheung, J. Chakareski, and Y. Ji, "Multiple description coding and recovery of free viewpoint video for wireless multi-path streaming," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 1, pp. 151–164, Feb. 2015.

[28] J. Chakareski, "Wireless streaming of interactive multi-view video via network compression and path diversity," *IEEE Trans. Commun.*, vol. 62, no. 4, pp. 1350–1357, Apr. 2014.

[29] J. Chakareski, "Transmission policy selection for multi-view content delivery over bandwidth constrained channels," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 931–942, Feb. 2014.

[30] J. Chakareski, V. Velisavljević, and V. Stanković, "User-action-driven view and rate scalable multiview video coding," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3473–3484, Sep. 2013.

[31] G. Cheung, V. Velisavljević, and A. Ortega, "On dependent bit allocation for multiview image coding with depth-image-based rendering," *IEEE Trans. Image Process.*, vol. 20, no. 11, pp. 3179–3194, Nov. 2011.

[32] C. Dorea and R. L. de Queiroz, "Attention-weighted texture and depth bit-allocation in general-geometry free-viewpoint television," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 5, pp. 1055–1065, May 2017.

[33] A. Hamza and M. Hefeeda, "Energy-efficient multicasting of multiview 3D videos to mobile devices," *ACM Trans. Multimedia Comput.*, vol. 8, no. 3s, p. 45, 2012.

[34] J. Chakareski, V. Velisavljević, and V. Stanković, "View-popularity-driven joint source and channel coding of view and rate scalable multi-view video," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 3, pp. 474–486, Apr. 2015.

[35] S. Kodera, T. Fujihashi, S. Saruwatari, and T. Watanabe, "Multi-view video streaming with mobile cameras," in *Proc. IEEE Global Commun. Conf.*, Austin, TX, USA, Dec. 2014, pp. 1412–1417.

KASHIF BILAL received the Ph.D. degree from North Dakota State University, USA. He is currently a Post-Doctoral Researcher with Qatar University, Qatar. He is also an Assistant Professor with the COMSATS Institute of Information Technology, Pakistan. His research interests include cloud computing, energy-ef?cient high-speed networks, and crowdsourced multimedia. He is awarded the CoE Student Researcher of the year 2014 based on his research contributions during his doctoral studies at North Dakota State University.

AIMAN ERBAD received the Ph.D. degree from British Columbia University, Canada. He is currently an Assistant Professor with the Computer Science and Engineering Department, Qatar University, Qatar. He was a participant in Qatar Leadership Centre which is program established by H.H. The Emir of Qatar to train leaders with high potential. He received the Platinum award from H.H. The Emir of Qatar Sheikh Tamim bin Hamad Al Thani at the Education Excellence Day (Ph.D. Category). He regularly serves as a Technical Program Committee Member in international conferences related to multimedia systems and networking (ACM Multimedia, ACM Multimedia Systems, NOSSDAV, and MoVid). He regularly consults local government agencies and research labs in issues related to networking, cloud computing, and the effects of technology on society.

MOHAMED HEFEEDA received the B.Sc. and M.Sc. degrees from Mansoura University, Egypt, in 1994 and 1997, respectively, and the Ph.D. degree from Purdue University, USA, in 2004.

He is currently a Professor with the School of Computing Science, Simon Fraser University, Canada, where he leads the Network Systems Laboratory. His research interests include multimedia networking over wired and wireless networks, mobile multimedia, and cloud computing. In 2011, he was one of the receipts of prestigious NSERC Discovery Accelerator Supplements, which are granted to a select group of distinguished researchers from all Science and Engineering disciplines in Canada. His research on mobile video streaming has resulted in multiple patents and conference awards (e.g., the ACM MMSys Best Paper, the ACM Multimedia Best Demo, and the IEEE Innovation Best Paper), and has been featured in several news venues, including ACM Tech News, World Journal News, and CTV British Columbia. (Check our News page for more info and links.)

He has co-authored over 100 refereed papers and multiple granted patents (Google Scholar Page). He was the Co-Chair or the Chair of several international conferences such as ICME, NOSSDAV, MoVid, and ACM Multimedia-Systems Track. He serves on the editorial boards of premier journals such as the *ACM Transactions on Multimedia Computing, Communications and Applications*, where he was named the Best Associate Editor in 2014. He maintains strong ties with industrial partners such as CBC, Aljazeera, Nokia, and Cisco, where some of his research software systems have been used.

In 2010, he co-founded Video Semantics in the U.S., which received over 1M in funding from the National Science Foundation and other agencies and its technology was later acquired.