IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 13, 2017, accepted October 13, 2017, date of publication November 1, 2017, date of current version December 5, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2768798

CoolConferencing: Enabling Robust Peer-to-Peer Multi-Party Video Conferencing

WEIMIN WU[1], HANZI MAO[2], YI WANG[1], JI WANG[1], WENKAI WANG[1], AND CHEN TIAN[3]

[1]School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan 430074, China
[2]Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77843-3112, USA
[3]State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China

Corresponding author: Yi Wang (e-mail: ywang@hust.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFC0806202, in part by the National Science and Technology Major Project of China under Grant 2017ZX03001013-003, in part by the Fundamental Research Funds for the Central Universities under Grant 0202-14380037, in part by the National Natural Science Foundation of China under Grant 61772265, Grant 61602194, Grant 61402198, and Grant 61321491, in part by the Collaborative Innovation Center of Novel Software Technology and Industrialization, and in part by the Jiangsu Innovation and Entrepreneurship (Shuangchuang) Program.

ABSTRACT Multi-party video conferencing (MPVC) is the next big opportunity for Internet streaming. Commercial MPVC solutions are either server-based or peer-to-peer (P2P)-based, which both have performance limitations. P2P technology is expected to dominate the MPVC platform. There are four requirements for a robust MPVC system: 1) realistic network assumptions; 2) realistic system settings; 3) multi-rate support; and 4) any-view support. Existing academic works study the problem from a theoretical perspective, and none of them meets all four requirements simultaneously. We design CoolConferencing, an overlay network for robust P2P MPVC. The core operations follow the easy-to-implement, robust, and resilient data-driven principle, which does not maintain complex global structures such as dissemination trees and can adapt to network dynamic distributedly and quickly. CoolConferencing is a robust system that meets all four requirements simultaneously. In addition, to the best of our knowledge, there is no existing work which examines its MPVC approach under various realistic network environments. We have evaluated CoolConferencing via an event-driven simulation. Compared with state-of-the-art video conferencing solutions, CoolConferencing achieves around 25% gain than Mutualcast and around 9% gain than Celerity in performance. Moreover, when the helper mechanism is enabled, CoolConferencing can easily exploit all available bandwidth to get optimal video transmission performance.

INDEX TERMS Computer networks, streaming media, peer to peer computing.

I. INTRODUCTION

A. MOTIVATION

Multi-party video conferencing (MPVC), which involves more than two participants in a live video conferencing session, is the next big opportunity for Internet streaming [1]–[5]. For a long time, due to its stringent bandwidth and delay requirements, MPVC has been limited to business market with dedicated equipments and network settings. Only recently, commercial platforms start to offer MPVC services to end-consumers; the leading providers are Skype Video Calls [6], Google+ Hangout [7] and Apple iChat [8].

Peer-to-peer technology is expected to dominate the MPVC platform. Existing commercial MPVC solutions are either server-based or directly-peered, which both have performance limitations. The effectiveness of peer-to-peer (P2P) Internet streaming has been proven by the success of many live streaming (Live) [9]–[11] and video-on-demand (VoD) systems [12]–[14]. P2P based systems can dramatically reduce the server load, and are ideal platforms for scalable content distribution.

In this paper, we demonstrate the design of a robust peer-to-peer multi-party video conferencing system.

B. CHALLENGE

Unlike Live or VoD streaming, in video telephony (both two-party and multi-party), the users' Quality-of-Experience (QoE) could degrade significantly if the one-way video delay goes over 350 milli-seconds [15]. Unlike two-party video chat, a MPVC session has many concurrent

video sources, and each participant may watch videos of all other participants; the required system capacity can be quadratically proportional to the number of participants [1], [4].

To build a robust MPVC system, there are four requirements that need to be considered:

- *Realistic network assumptions.* The first issue is network bottleneck. Many P2P analytic studies only consider the cases where bandwidth bottlenecks reside at the edge of the network, *i.e.*, uplinks or downlinks of end systems. While in practise, bottlenecks can appear anywhere in the network core [16], [17]. The second issue is network delay. Many existing studies only control the number of propagation hops, but ignore the explicit delay of the underlying network. However, one-way delay between two overlay nodes can easily go beyond hundreds of milli-seconds [18], which is not negligible for viewing experience. Even worse, both network bandwidth and delay are dynamic in Internet.
- *Realistic system settings.* A P2P system should have a efficient utilization of network resource to optimize user experience; still, it is not uncommon that in many situations, the sum of participants' uplink bandwidth falls short to meet the content demand. In current P2P streaming networks, the helper mechanism is widely used [19]–[21]. A *helper* is a special peer, which receives videos streams for the purpose of helping the session; it is not a participant of the ongoing video conference; it can be either an idle video conference client, or a dedicated server provided by the service provider. Due to highly stringent capacity requirements, helpers are even more important in MPVC than in Live and VoD.
- *Multi-rate support.* In practice, the requirement of streaming rate varies across users, especially for mobile users [22], [23]. For example, a mobile terminal may not need a high resolution playback due to the restricted screen size, even though the access link is abundant in bandwidth. Also, since WiFi hotspots are crowded usually, and cellular networks provide low bandwidth, high-bandwidth wireless/cellular access sometimes is really rare for video streaming. Moreover, a mobile user may also choose to reduce the data rate due to limited usage quota enforced by the operator contract.
- *Any-view support.* To date, the dominant MPVC viewing mode is still *all-view*, whereby every participant watches videos of all others. Without the freedom of choices, all participants are viewed equally and cause the same bandwidth. As a result, Skype Video Call suggests to limit the number of participants to lower than 5. Note that users have subjective preference of video quality [24]. We argue that a system should respect the most flexible user choices, such as choosing arbitrary number of high quality participants, and completely close some participants' video; we name this mode *any-view*.

There exists several academic works for P2P-based MPVC recently [25]–[29]. However, most related works study the problem from a theoretical perspective. None of them meets all four requirements simultaneously (see § II-B for details).

C. CONTRIBUTIONS

To this end, we design CoolConferencing, an overlay network for P2P multi-party video conferencing. The core operations in CoolConferencing follow the data-driven principal [10]: a mesh topology is established among participants for each session; every participant periodically exchanges video data availability and status information with others; a participant retrieves desired video data from, and supplies data to, others. The data-driven design has two well-recognized advantages: (1) *easy-to-implement*, as it does not maintain complex global structures such as dissemination trees; and (2) *robust and resilient*, as our solution is purely distributed, without single point of failure which can adapt to network dynamic distributedly and quickly, and receivers can adaptively switch among multiple possible data suppliers based on the up-to-date data availability and status information.

CoolConferencing is a practical system that meets all four requirements simultaneously. Also, to our best knowledge, there is no existing work which examines its MPVC approach under various realistic network environments. For Live and VoD, Liang *et al.* have exploited the balance between the P2P streaming performance and the design complexity [30]. They conclude that the performance is insensitive to scheduling when the streaming rate is low and long playback delays are tolerable, and P2P streaming performance becomes highly sensitive to scheduling when the streaming rate approaches the maximum supportable rate of the system and the tolerable delays become small. As we have mentioned above, MPVC is more demanding in both delay and bandwidth requirement.

We have evaluated CoolConferencing via an event-driven simulation. Compared with a state-of-the-art video conferencing solution, CoolConferencing can achieve around 25% gain over Mutualcast [29] and 9% gain over Celerity [27] when supply ratio of the system equals to 1. Moreover, when the helper mechanism is enabled, CoolConferencing can easily exploit all available bandwidth to get optimal video transmission performance.

We describe the general architecture and important building blocks of CoolConferencing (Section III). The peer design is discussed (Section IV). By extensive evaluations, we demonstrate that CoolConferencing can deliver close to optimum performance under various network environments (Section V). Finally we conclude the paper (Section VI).

II. BACKGROUND

In this section, we present backgrounds of P2P Internet streaming and discuss unique challenges of multi-party video conferencing. The related industry and academic works are reviewed and compared with CoolConferencing.

A. INTERNET STREAMING

1) LIVE AND VoD

Online streaming is already dominating the traffic in today's Internet. Currently, two major applications of streaming are

live-streaming (Live) and video-on-demand (VoD). Each Live/VoD session has only one video source, a large session of which can have tens of thousands peers online simultaneously.

For scalable content distribution, many streaming networks have become peer-to-peer (P2P) based [9], [10], [12]. Most designs are BitTorrent-like data driven approaches: guided by a tracker, peers (viewing the same video) connect with some neighbors to form an overlay mesh network to relay data. A source encodes small clips of video data called pieces; a peer can download pieces from the source directly; or a peer can also exchange piece bitmaps with a neighbor so that they can request pieces from each other.

The key data structure of a streaming application is its content buffer. Taking a peer in a Live session as an example, its buffer status at time $t = 100$ is shown in Figure 1. A shaded piece is one that has been downloaded; otherwise the piece is still missing. The rightmost piece of the buffer is the most recent piece produced by the source, and we refer to this piece as the *sourcepoint*. Another important piece at this moment is the next piece to be delivered to the media player; we refer to it as the *playpoint*. For VoD, there is no sourcepoint, since all content are already available. A main metric of streaming performance is piece missing ratio, which is the fraction of pieces that are not received by the playback deadline.

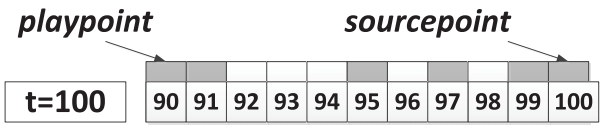


FIGURE 1. Content buffer.

2) MULTI-RATE SUPPORT

Multi-rate support is necessary to deal with the inherent heterogeneity among streaming users. For example, a user inside an ADSL network usually has much lower uplink and downlink bandwidth compared with another user in an enterprise network. Another reason may be user requirement: a user with an iPhone has much smaller screen size compared with normal users; receiving high resolution streaming is no more than a waste in this case.

There are two common multi-rate coding approaches: Scalable Video Coding (SVC) [31] and Multiple Description Coding (MDC) [32]. SVC encoder generates one base video layer and several enhancement layers; the base layer is required for decoding in all receivers; the enhancement layers are optional and can be received to improve video quality. As a comparison, MDC generates multiple substreams, and each substream can be independently decoded; the more substreams received, the better the video quality. Multi-rate coding schemes all have coding overhead: compared with single-rate coding for the same quality, the overhead of SVC can range from 10% to 50% with different configurations, and MDC has an overhead of 30% to 40% [31], [33]–[35].

Another multi-rate approach is Partitioning, which is widely used for dynamic streaming [36]. A single source is encoded to several versions with different qualities and rates; specific receivers are partitioned to different groups according to their received versions; only receivers in the same group can help each other, as content are different among groups. Evidently, the coding overhead of single-layer Partitioning scheme are negligible compared to the SVC and MDC.

3) MPVC UNIQUE CHALLENGES

Multi-party video conferencing has two unique challenges. First of all, the delay requirement is more stringent. In video telephony, users' QoE degrades significantly if the one-way video delay goes over 350 milli-seconds [15]. As a comparison, Live normally can tolerates tens of seconds of content delay [10].

Secondly, compared with other streaming, MPVC is much more bandwidth-demanding.. A MPVC session has multiple concurrent video sources, and each receiver may watch videos of all other participants. For Live and VoD, the required system capacity increases linearly with the number of peers; while for MPVC, the required capacity is increasing quadratically.

B. MULTI-PARTY CONFERENCING SYSTEMS

Recently, the increasingly faster residential network accesses gradually pave the way for end-consumers' MPVC services. Commercial platforms start to offer MPVC services to end-consumers [1]–[5]; there also exists several academic works for P2P-based MPVC [25]–[29].

Commercial MPVC solutions can be classified to two categories, and both have limitations. The first category is largely server-based: a participant chooses a dedicated server as his proxy, and uploads his video data to the proxy; the video from other participants are firstly aggregated by the proxy and then downloaded by this participant; Skype Video Calls and Goolge+ Hangout fall in this category. It is well known that, sever-based conferencing solutions are not economically scalable, hence their prospect of free or low price service provision remains doubtful (if not completely impossible). The second category is Simulcast: each participant splits his uplink bandwidth equally among all receivers and streams to each receiver separately; Apple iChat falls in this category. Simulcast is simple to implement, but tends to produce unsatisfactory quality: participants with low uplink bandwidth have to encode with low video rate and this significantly degrades the perceptual experience of other users.

For existing academic solutions, so far none of them considers all robust issues mention in § I simultaneously. Table 1 summarizes some related work. Specifically, Ponec *et al.* [25], [26] support multi-rate video conferencing via SVC techniques. Assuming uplink is the only bottleneck, they develop optimal tree packing algorithms for the multi-rate multi-cast problem. They do not explicitly handle the delay constraint, while instead limit the tree depth to

TABLE 1. Summary of previous approaches and comparison to CoolConferencing.

	Network Bottleneck			delay	Helper	Multi-Rate		User Interface		
Related Work	Uplink	Path	Downlink	Explicit	Explicit	SVC/MDC	Partitioning	All-View	One-View	Any-View
Ponec *et al.* [25, 26]	Yes	No	No	No	Yes	Yes	No	Yes	No	No
Zhao *et al.* [28]	Yes	No	No	No	Yes	No	Yes	No	Yes	No
Kurdoglu *et al.* [29]	Yes	No	Yes	No	No	Yes	Yes	Yes	No	No
Celerity [27]	Yes	Yes	Yes	Yes	No	No	No	Yes	No	No
CoolConferencing	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

two hops. Zhao *et al.* [28] is similar in terms of network assumption. The unique feature of this work is the one-view mode support, where each participant can choose one user to watch at high video quality, and watch all other participants' videos at the minimum video quality; in this case the multi-rate approach is actually receiver-partitioning. Kurdoglu *et al.* [29] considers both uplink and downlink network bottlenecks, however, packet loss and transmission delay is ignored in the design of the system. The work closest to CoolConferencing is Celebrity [27]. The solution handles both bandwidth and delay aspects explicitly for a MPVC design. Based on distributed rate control protocol, Celebrity can adapt to topologies and network conditions quickly. However, multi-rate and helper support are not considered by the project.

To our best knowledge, CoolConferencing is the first system that simultaneously considers all robust issues. Shown in Table 1, none of existing work supports flexible any-view mode. An additional difference is that: all existing solutions are tree based, which is less robust and resilient than the data-driven approaches. There are also loosely-related works. Liang *et al.* [37] study the optimal bandwidth sharing across multiple MPVC sessions. Feng *et al.* [38] propose that video conferencing should take the advantage from the inter-datacenter network in the cloud. Their contributions are orthogonal to CoolConferencing.

III. OVERVIEW AND BUILDING BLOCKS

A. SYSTEM OVERVIEW

Similar to other P2P streaming systems, CoolConferencing has the following logic components: (1) trackers to help peers connect to other peers; (2) sources generate conference streaming; (3) receivers as the consumer of streaming; (4) suppliers which relay the streaming for sources.

There are two kinds of peers. The first is *participant*. A participant can act as a source, a receiver, or a supplier, and mostly all of these roles. Each participant has an any-view user interface. This interface enables a user to arrange other participants to three groups according to his preference: Participant Important (*PI*), Participant Normal (*PN*) and invisible, an example of which is shown in Figure 2. Assume there are 6 participants A, B, C, D, E, F in a MPVC session, and the screens of F and D are demonstrated. F chooses to watch all other 5 participants; F selects A and B into the *PI* group (with high quality video) and leaves the rest to the *PN* group (with low quality video); as a comparison, D only sets A as *PI*, and closes the videos of B and E.

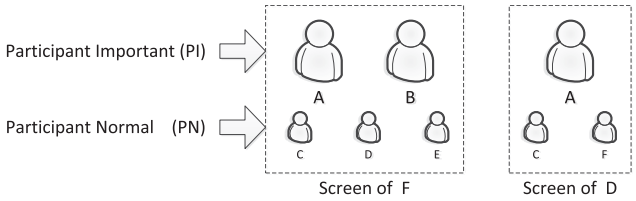


FIGURE 2. Examples of any-view.

The second is *helper*. Note that a helper doesn't need a user interface. Instead, a special helper algorithm decides which data it should receive and forward to help the session (§ IV-D).

In this part, we first introduce the topology construction among peers (§ III-B); the data delivery protocol is presented in § III-C; multi-rate support is discussed in § III-D; finally the update protocol is defined (§ III-E).

B. OVERLAY TOPOLOGY

In CoolConferencing, a fundamental decision is the topology of each session's overlay. Due to that the numbers of users in a Live/VoD session are usually large, it is impossible for each node to connect to every other node; as a result, their topologies are usually mesh. However, a MPVC session is in general small in size, as the number of participants seldom goes beyond 10 - 15 [25]. Exploiting this feature, CoolConferencing chooses a hybrid structure: a full-mesh for network information exchange such as the neighbor buffer map and delay map, and a mesh for the delivery of user data. Due to that the network delay among peers may be large and the users' video streams are delay sensitive, CoolConferencing cannot use full-mesh to deliver data packets, as that may violate the playback deadline. However, signal packets are not deadline-sensitive; with a full-mesh based information exchange , each peer can have a global view of the whole content overlay. The benefit is to improve the efficiency of peers' distributed scheduling decisions (§ IV).

C. DATA DELIVERY

There are two data delivery approaches: *pull* and *push*. In pull mode, each peer updates its piece buffermap to the network; a receiver issues a request for a piece; the supplier sends the piece. In push mode, a receiver subscribes a layer/substream/version to a supplier, and the new streaming data pieces are sent directly to the receiver by the supplier, as soon as it is generated or received from another peer.

P2P Live and VoD systems mostly use pull mode. However, its additional content delay is roughly three times of one-way delay between two peers: one for buffermap updates, one for piece request, and the last for piece send. As a comparison, push-based delivery can have a delay as low as just one time of one-way delay. Due to the stringent delay requirement, CoolConferencing chooses the push-based data delivery. Similar to existing streaming platforms, we use UDP to send the pieces, and rely on application level congestion control to modulate the traffic.

D. MULTI-RATE

CoolConferencing uses a unified framework to support three multi-rate approaches. Specifically, we adopt the abstraction *channel* to denote: either a layer in SVC, or a substream in MDC, or a rate version in Partitioning. Each source publishes the availability of channels; each receiver sends subscription to selected channels, based on user's any-view preference. For simplicity of presentation, in this paper we assume two quality levels for each approach.

Specifically, each SVC source has two layers (L and H); from the perspective of a receiver, we use $PI(L)/PI(H)$ to denote the base layer and the enhancement layer of a PI; while for a PN, only the the base layer $PN(L)$ is requested. Correspondingly, in our example, MDC has two substreams (0 and 1), and Partitioning has a low rate version and high rate version (L and H).

An important problem is: how to define priority among channels to solve resource competition? The rationale behind the priority definition is that: (1) it is more important to make all specified participants visible, compared with only PI visible but with high quality; hence $PI(L), PN(L) > PI(H)$; and (2) since $PI > PN$, it is reasonable to set $PI(L) > PN(L)$. Start from here, for SVC, the priority defined by CoolConferencing is $PI(L) > PN(L) > PI(H)$. This is also consistent with SVC coding characteristics: enhancement layer is useless without the base layer [39].

MDC is more complicated: there is no semantic priority among substreams. For example, for a PN source, a receiver can choose either substream 0 or 1, both achieve the same quality; for a PI, a receiver needs to subscribe both. The problem is that: random selection of substreams from the same source hurts the sharing opportunity in a P2P system (demonstrated later in § V-B). CoolConferencing instead requires each source to pre-define a priority to all substreams, then priority rule similar to SVC can be derived.

Partitioning is even more complicated, since its low version and high rate version are mutually exclusive. The priority defined by CoolConferencing is $PI(L) > PI(H), PN(L)$. When competing for bandwidth, $PI(H)$ and $PN(L)$ should be treated equally; however, a receiver would downgrade to $PI(L)$ from $PI(H)$ only if it cannot receive a stable high quality version of a PI; for this reason, CoolConferencing gives high priority to $PI(L)$.

The priority of bandwidth is enforced in suppliers' uplink scheduling (§ IV-C). We believe these designs can be easily extended to support more complex scenarios, such as more-than-two-channel multi-rate schemes, or with different priority rules.

E. UPDATE

An update message between two nodes contains important information. The first information is node's buffermap, which is used to describe the delay of the channels and contains both the generated and the currently subscribed channels (from other participants). Together with each channel, the content delay is also attached. Since pieces are sequentially pushed from a supplier to a receiver, content delay can instead be denoted by the largest index of the received pieces (Figure 3). With this information, the delay control of a receiver (for supplier selection of channels) is more convenient.

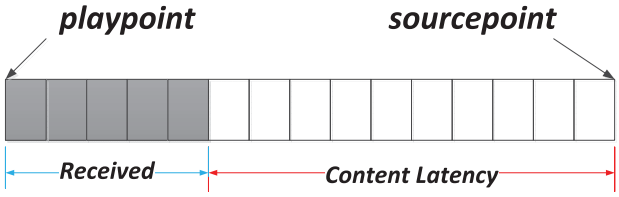


FIGURE 3. Content delay.

The second information is the delay map which contains the delays between each pair of nodes. A node module monitors the one-way delay between the node and all other peers (Section IV); the purpose is also to help delay control. The third information is downlink channel piece missing ratio, which is defined as the ratio between the pieces received at a certain peer and the pieces sent by its supplier, and denoted by DH for notational brevity. The fourth information is an uplink piece missing index denoted by UH, which reports the estimated uplink bandwidth. A peer's UH is defined as the average of the UH of its receivers.

IV. PEER DESIGN

A. SYSTEM DIAGRAM

Figure 4 depicts the system diagram of a CoolConferencing node. There are four key data components: (1) content buffer, which stores both the data of generated video channels by the user himself, and the data of channels received from other participants; (2) node buffermap, which maintains and updates information of buffered (generated and received) channels; (3) neighbor buffermap, which receives and maintains buffermap information from other participants; (4) delay map, which contains the measured delay between the node and other participants. There are three key modules.

1) PARTICIPANT MANAGER

Participant Manager maintains the information of all other participants in the session. A key function here is to generate the delay map.

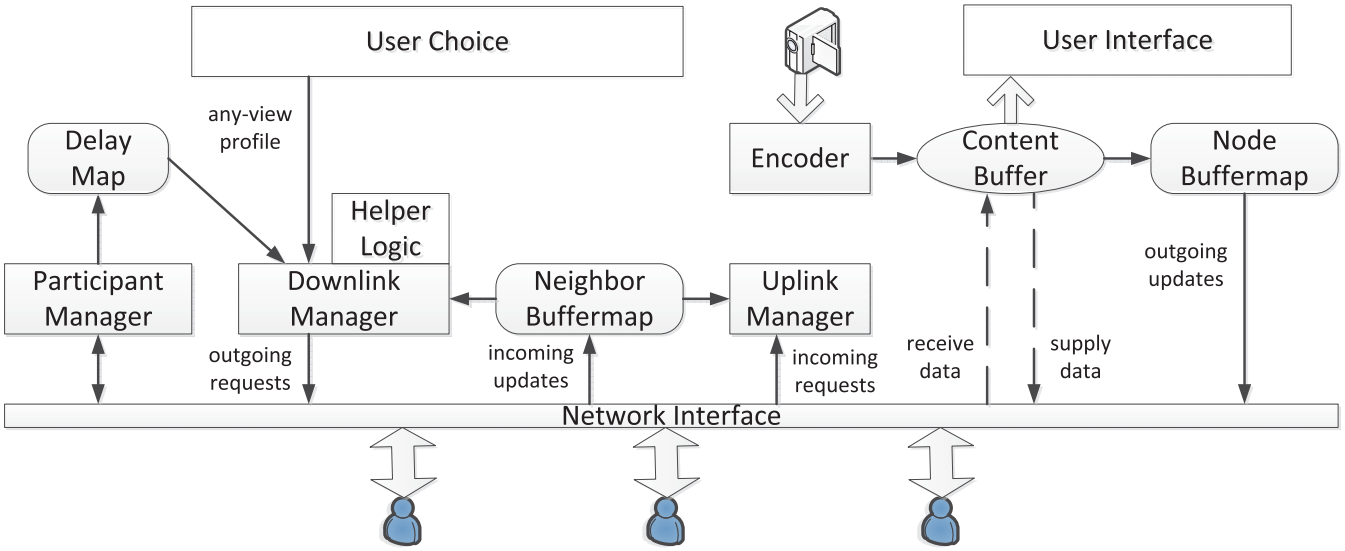


FIGURE 4. System diagram of a peer.

2) DOWNLINK MANAGER

There are many roles for Downlink Manager. First, based on receiver's preference of sources and current network status, it chooses which of source's channel to request. Second, for each selected channel, there usually exists a number of suppliers. Downlink Manager selects, based on neighbors' update information, from which supplier to request. Downlink Manager also periodically scans for better suppliers for an already-subscribed channel for the purpose of better performance. Lastly, Downlink Manager needs to maintain downlink lossy index DH for each channel.

3) UPLINK MANAGER

Uplink manager has three functions. The first is to estimate uplink bandwidth based on the current congestion level and derive uplink piece missing index UH. The second is to perform prioritized uplink scheduling based on estimated bandwidth. The third function is to perform admission control based on UH.

B. DOWNLINK ALGORITHMS

1) SELECT CHANNEL

Initial channel selection is straightforward for SVC: for PI, both $PI(H)$ and $PI(L)$ are selected; for PN, only $PN(L)$ is selected. Given source determined substream priority, MDC channels can be selected in a similar way. For Partitioning, since H and L are exclusive, $PI(H)$ will be selected first for a PI; if after a certain period, $PI(H)$ cannot obtain stable performance, Downlink Manager cancels $PI(H)$ and turned to $PI(L)$.

2) SELECT SUPPLIER

Above all, the playback deadline T_j of channel j should be respected. Algorithm 1 lists the pseudo code of supplier selection. For a receiver n, each supplier i's channel delay L_{ij} which stands for the delay of the channel when arriving at this supplier, and nodes' delay map information D_{ni} are used together to get a set of the candidate suppliers set C (Line 2 to Line 6). Then the candidate set is sorted by examining the suppliers' uplink lossy status UH_i (Line 8), and the one with the minimum UH_i is returned.

Algorithm 1 Supplier Selection

```
1:  Procedure SelectSupplier(L_ij,D_ni,T_j)
2:  for all supplier i do
3:      if L_ij + D_ni < T_j then
4:          C ← i(put i in the candidate suppliers set C)
5:      end if
6:  end for
7:  for all i ∈ C do
8:      increasingly sort all the suppliers in terms of UH_i
9:  end for
10: return i' with the minimum UH_i
11: end procedure
```

C. UPLINK ALGORITHMS

1) UPLINK BANDWIDTH ESTIMATION

Downlink lossy status DH_{ij} report from receivers i is basically the piece missing ratio of channel j. If only a small fraction of receivers report missing ratio, it is an indication that there are network bottlenecks in the path from n to the receivers. Otherwise, all receivers report high missing ratio, then it is a clear indication that supplier n has overloaded its uplink bandwidth. In this case, Uplink Manager set uplink lossy index UH_n to be the average of all DH_{ij} reports.

Accordingly, Uplink Manager also adjusts its bandwidth estimation. Let BW_t denotes the uplink bandwidth

estimation in time t, Algorithm 2 lists the pseudo code. Basically, CoolConferencing emulates DCTCP [40] behaviour: when UH_n is zero, estimation is increased; otherwise, it reduces proportionally to the piece missing ratio.

Algorithm 2 Uplink Bandwidth Estimation

```
Procedure UplinkEstimation(UH_n, BW_t)
if UH_n = 0 then
    BW_{t+1} = BW_t + 1
else
    BW_{t+1} = BW_t(1 − UH_n)
end if
return BW_{t+1}
end procedure
```

2) UPLINK SCHEDULING

Each supplier maintains three priority queues; when a new piece is generated or received from other participants, it is duplicated for forwarding to subscribed receivers; these pieces enter the corresponding priority queue first; based on the estimated uplink bandwidth BW_t, periodical paced sending is performed in the supplier; the dequeue sequence follows strict priority ($green > yellow > red$).

Shown in Figure 5 is a SVC example: receiver C sets A as PI and B as PN; receiver D sets B as PI and A as PN; E sets both A and B as PI,and it is the supplier of both A and B. E receives $A(H)/A(L)$ and $B(H)/B(L)$ from A and B respectively; as a supplier, it forwards $A(H)/A(L)/B(L)$ to C and $A(L)/B(H)/B(L)$ to D. According to the channel subscription update from receivers, E decides that $A(L) \rightharpoonup C$ and $B(L) \rightharpoonup D$ should enter the green queue; $A(L) \rightharpoonup D$ and $B(L) \rightharpoonup C$ should enter the yellow queue;$A(H) \rightharpoonup C$ and $B(H) \rightharpoonup D$ should enter the red queue.

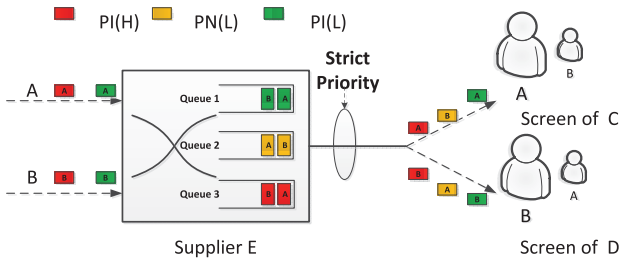


FIGURE 5. Scheduling with queue.

MDC approach uses a similar setting. As mentioned above, Partitioning should be different in uplink scheduling. In this case, only two queues are deployed in a supplier: a green queue for $PI(L)$ and a yellow queue for $PI(H)$ and $PN(L)$ together.

3) ADMISSION CONTROL

Normally, every incoming push request is accepted when UH_n is zero or low; otherwise if UH_n is higher than a threshold, all new requests are directly rejected.

D. HELPER ALGORITHM

In a helper node, the helper logic module is activated in the Downlink Manager. When a helper still has available uplink bandwidth, it dynamically picks a channel to receive and forward for the session based on the criteria in the following. Specifically, CoolConferencing has developed two intuitive channel picking algorithms: (1) *Worst First*, where the helper monitors the whole session, and selects the channel with the worst service quality; (2) *Random*, where the choice is made randomly.

Note that helper always request from source, and we enforce an extra priority (over all other traffic) for helper traffic in source uplink scheduling.

V. EVALUATION

In this section, we evaluate the different aspects of CoolConferencing. The evaluation is performed in a packet-level event-driven simulator written in C++, in which part of the code are from our previous live streaming platform [11]. We set the playback mode to 40 pieces per second; each experiment runs for 30 minutes. Note that we do not explicitly set the rate of channel; it is normalized to the number pieces per second, hence we can emulate arbitrary streaming rate by specify the size of each piece.

Throughout the paper, we use a toy topology (shown in Figure 6) as a default setting, where each link has a network delay of 50 ms. The conferencing overlay contains 12 participants, and the delay between each pair of nodes is dominated by the shortest physical path (*e.g.*, 250 ms between peer 1 and 12). The major control parameter is *supply ratio*, which is the ratio between the total participants' upload capacity and the total demand of streaming bandwidth. In this topology, the upload capacity is randomly allocated to each peer.

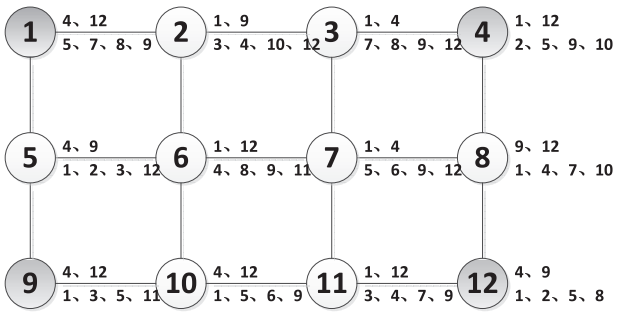


FIGURE 6. An examples of Topology with any-view.

We set the play-back deadline to 350 ms, thus not every multi-hop path can satisfy the deadline (*e.g.*, $1 \rightarrow 12 \rightarrow 2$). Without explicit note, two-layer SVC is used as the default multi-rate scheme, where layer 0 is the base layer, and layer 1 is the enhancement layer; we assume all layers have the same rate for simplicity; the default supply ratio is set to 1. We select 4 nodes (1, 4, 9, and 12) as sources that can serve high-quality streaming. For each participant, two high-quality PIs

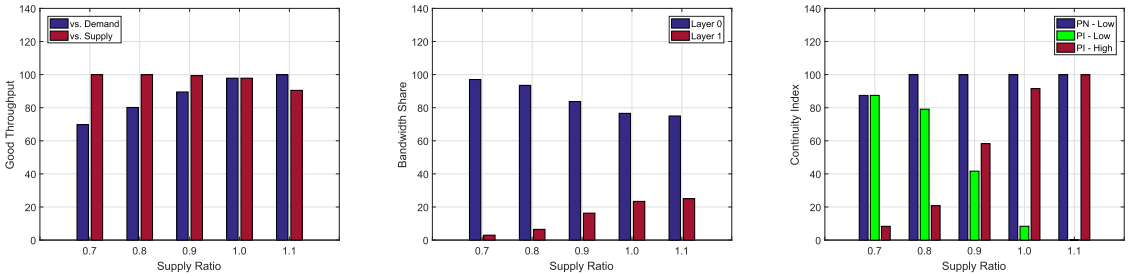


FIGURE 7. CoolConferencing under stable environment (a) Good Throughput vs. supply ratio; (b) Bandwidth share between layer 0 and 1; and (c) Continuity Index comparison of PI and PN.

and four low-quality *PN*s are randomly selected as his any-view preference (*e.g.*, 4 and 12 are *PI*s for node 1). In this setting, for a normalized demand 1 in SVC, 75% demand should go to layer 0, and 25% should go to layer 1: each *PI* needs one *H* and one *L*; each *PN* needs one *L*; hence for each viewer, the ratio between *H* and *L* is 2 v.s. 6.

The following performance metrics are used: (1) *good throughput*, which is an indication of uplink network resource utilization; (2) *bandwidth sharing*, which is the resource usage split among layer 0 and layer 1 channels; and (3) *continuity index*, which is the ratio between the pieces played and the total number of pieces within the deadline of 350ms (*i.e.*, 1 minus piece missing ratio).

Summary of Results:

- We show that CoolConferencing can deliver close to optimum performance under stable network environments, with the presence of bottlenecks in network core, and under dynamic network scenarios, such as variation of link, node and user any-view preference(§ V-A.1).
- We evaluate important components, and demonstrate that fine-tuned helper algorithms can deliver significant better performance than a naive random solution, priority mechanism in uplink scheduling is critical, and different multi-rate approaches have their advantages in different scenarios (§ V-B).
- We evaluate the performance of the CoolConferencing compared to the Mutualcast and layered distribution-based P2P video conferencing systems in [29] and the single-rate scheme Celerity in [27], and validate the advantage of the proposed scheme. we can observe that the proposed CoolConferencing outperforms the Mutualcast [29] and Celerity [27] in the whole region of supply rate. And as the supply ratio decreases, the gains over Mutualcast [29] become more and more significant. Moreover, it is shown that when the helper is enabled in CoolConferencing, the gain over Mutualcast [29] and Celerity [27] can be even more manifest(§ 5.3).

A. COOLCONFERENCING UNDER VARIOUS ENVIRONMENT

1) STABLE ENVIRONMENT

We start by evaluating CoolConferencing in stable environments. The supply ratio is increased from 0.7 to 1.1. In Figure 7, the results are averaged over the data of all nodes.

Figure 7(a) demonstrates the ratio of total good throughput versus the demand and supply. When supply is less than or equal to demand (*i.e.*, supply ratio 0.7 to 1.0), the provided capacity is fully utilized: as shown in the figure, the throughput/supply ratio is always close to 100%; it only drops when bandwidth is over-provisioned (*i.e.*, supply ratio 1.1). It is clear that CoolConferencing can fully exploit the available bandwidth to meet the streaming demand.

Figure 7(b) presents the percentage of throughput used by layer 0 and 1: when supply is less than 0.75, even layer 0 demand cannot be fully met, hence almost all bandwidth is consumed by layer 0 in this case (supply ratio 0.7), due to its priority in uplink scheduling. With the increase of supply, layer 1 channels gradually gets more bandwidth: its share increases from nearly 0 (supply ratio 0.7) to nearly 25% (supply ratio 1.0 and 1.1), consistent with its share in demand.

The continuity index, which relates with participant's Quality-of-Experience (QoE) directly, is shown in Figure 7(c): with a supply ratio 0.7, PI and PN in average have a continuity index of 96%/87% respectively; when the bandwidth supply increases, all participants quickly approach 100% in continuity index.

We decompose the continuity performance to $PN-Low$, $PI-Low$ and $PI-High$ respectively, to further check the effects of multi-rate priority control; as mentioned above, here we assume *PN* only receive one layer and the quality is $PN-Low$; $PI-Low/PI-High$ are the quality of *PI* with one/two received layers separately.

As shown in Figure 8, the queue based scheduling faithfully enforces the priority: even with a supply ratio 0.7, *PI* can achieve a 95.77% continuity (8.33% are $PI-High$); this indicates that almost all *PI* are visible to the participants in such a constrained scenario. As a comparison, *PN* is around 87.44%, as it is inferior when compete with *PI*. When supply increases to 0.8, $PN-Low$ quickly increases to 99.9%, which indicates fully satisfied; the visible ratio of *PI* ($PI-High+PN-Low$) is also over 99.9%, while 20.82% is high quality. With the further increase of supply, more and more *PI* move to higher quality. Note that in Figure 8, we compute the Continuity Index for PI and PN separately. Thus, when the supply ratio is high, the percentage for (PI-Low+PI-High) is almost 100% and the percentage for PN is also 100%.

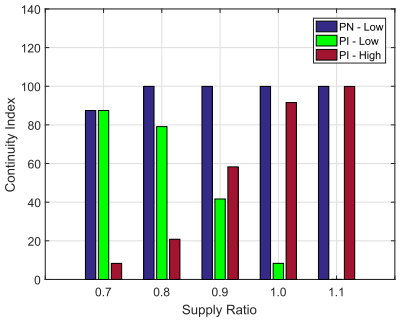


FIGURE 8. Performance comparison of continuity index.

To sum up, it is clear that CoolConferencing can deliver close to optimum performance under stable network environments.

2) BOTTLENECK IN THE NETWORK CORE

In this part, we first evaluate the impacts of in-network bottlenecks. Specifically, 10% of the overlay links are randomly chosen to have 0 capacity; other links are proportionally amplified to maintain the total supply ratio unchanged. The results (black) are shown in Figure 9(a): the performance of both layers are close to the results of stable scenario (red), which does not have bottlenecks. Our analysis of the experiment log verifies that *Select Supplier* module can avoid bottleneck links; also, *Uplink Bandwidth Estimation* correctly identifies the existence of link bottleneck, and the uplink bandwidth can be correctly estimated.

One step further, we evaluate CoolConferencing with several random topologies with network bottleneck. The results (green) in Figure 9(a) are even a little bit better than our default toy topology. The conclusion is that CoolConferencing can provide close to optimum performance even with the presence of bottlenecks in the network core.

3) DYNAMIC ENVIRONMENTS

In this part, we consider the impact of dynamics on system performance. There are three kinds of dynamics: link, node and user any-view preference.

In the first experiment, we set the uplink bandwidth of all nodes to be dynamically changing. More specifically, in the simulation, the uplink bandwidth for the nodes changes every 5 seconds. In the simulation, the dynamics of uplink bandwidth for all the nodes are changed dynamically according the ratio summarized in the Table 2. As shown in Figure 9(b), we can see that CoolConferencing can respond to bandwidth changes: as the uplink bandwidth decreases, continuity index for the system decreases accordingly; then the performance recovers after the bandwidth gets back to normal.

In the second experiment, we emulate node dynamics. In the example topology, peer 11 is not a high-quality source, and its demand and capacity happen to be similar. At 10 minutes, peer 11 is removed from the system, and it comes back at 20 minutes. As shown in Figure 9(c), since peer 11's dynamic does not affect the system's total supply ratio, the continuity

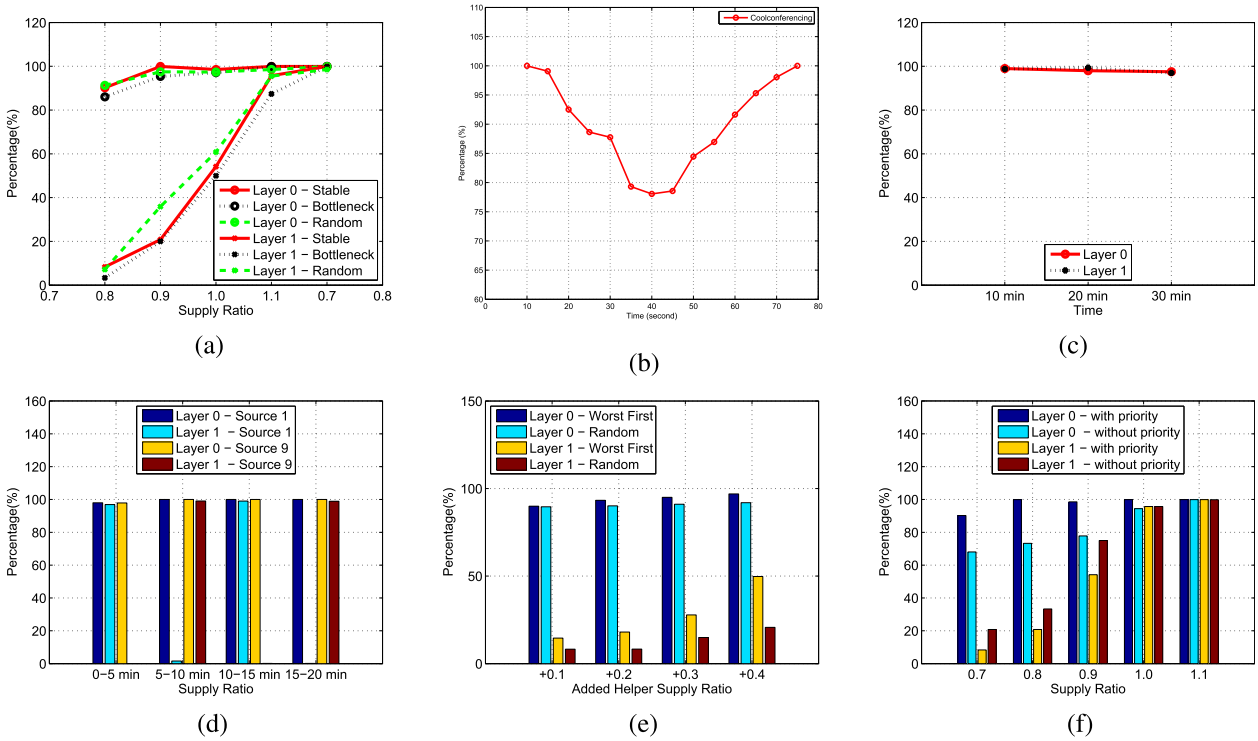


FIGURE 9. Continuity index of layer 0 and layer 1 in different scenarios. (a) Bottleneck and Random Topology. (b) Link dynamic. (c) Node dynamic. (d) Any-view dynamic. (e) Helper algorithms. (f) With/without Priority.

TABLE 2. Link dynamics configuration for uplink bandwidth.

Time (second)	10	15	20	25	30	35	40	45	50	55	60	65	70	75
Ratio of UL BW	1	0.95	0.9	0.85	0.8	0.75	0.7	0.75	0.8	0.85	0.9	0.95	1	1

index of both layers almost maintain stable for all the node dynamic events.

In the third experiment, we emulate the dynamic of user behaviours, *i.e.*, preferred any-view profile. More specifically, peer 7 alternatively selects node 1 and 9 as his preferred high-quality source, every 5 minutes. In the whole process, the system good throughput of both layers remain stable. Peer 7's continuity index of source 1 and 9 are shown in Figure 9(d): the transition between quality levels is smooth.

B. COMPONENT EVALUATION

1) HELPER ALGORITHM

In this part, we illustrate the effectiveness of different helper algorithms. We choose the most challenging scenario with supply ratio 0.7. When an experiment starts, we add a helper in 5 minutes. For each case, the added helper capacity is increased from 0.1 to 0.4 (total capacity increases to 0.8 until 1.1). Two algorithms are compared: *Worst First* and *Random*.

As shown in Figure 9(e), *Worst First* helper is significantly better the than *Random* algorithm. For example, when adding 0.3 supply ratio by helper, *Worst First*'s continuity index of layer 0 is 5% higher than that of the *Random* helper, and layer 1 is even 13% higher.

Note that our helper algorithm does not enforce priority of layer 0 over layer 1; when compared with a non-helper case with the same supply ratio, the results obtained are different: the with-helper case has lower performance for layer 0 and higher performance for layer 1.

2) PRIORITIZED UPLINK SCHEDULING

In this part, we analyze the effect of prioritized uplink scheduling on system performance. We intentionally disable the priority of layer 0 in uplink scheduling. Now all traffic have equal weights.

The results are shown in Figure 9(f). In the figure, without priority, layer 1's traffic can get a significantly higher share of bandwidth; as a consequence, layer 0's continuity index is significantly affected. The conclusion is that priority in uplink scheduling is critical.

3) COMPARISON OF SVC, MDC, AND PARTITIONING

In this part, we compare different multi-rate approaches. For MDC, in our setting, a receiver requests for all two substreams from a preferred *PI* source, and chooses substream 0 for a *PN* source. Partitioning simply follows the procedure defined by the *Select Channel* module. We set a control threshold for video quality: when a channel has a piece missing ratio larger than 10%, the channel cannot be decoded; the receiver needs to change from $PI - High$ to $PI - Low$.

Figure 10(a) shows the breakdown of playback time by number of experienced substreams under MDC. For example, when the supply ratio is 0.7, for a PI, a participant can only watch 2 substreams for 8.33% of the whole process (high quality); for 87.44% of total time, he can only watch 1 substream (low quality); and even worse, for 4.23% of total time, no substream from this PI can be viewed in a reasonable quality. With the increase of supply ratio, the user perceived quality also increases: when the supply ratio equals 1, there is almost no source getting dropped at all, although a small portion of preferred PI sources are viewed in low quality. The same goes for Partitioning. As shown in Figure 10(b), the percentage of full rate playback time is very low when the supply ratio is 0.7, and gradually increases with supply.

We have found that the bandwidth utilizations of MDC and SVC are higher than Partitioning. As shown in Figure 10(c), the utilization of total upload bandwidth can be as low as 75% in Partitioning approach. With layered coding (SVC/MDC), receivers receiving different subsets of layers can still share common layers: in SVC all receivers would at least share the

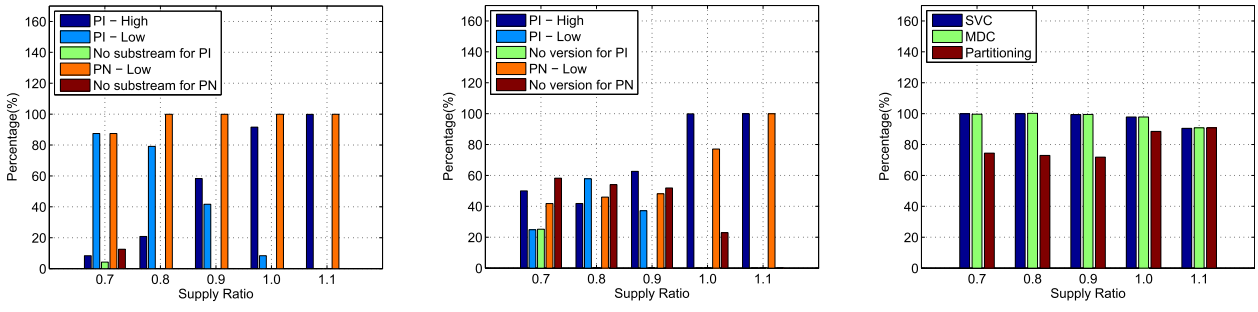


FIGURE 10. Comparison of the three multi-rate approaches in playback times. (a) MDC. (b) Receiver-partitioning. (c) Good Throughput vs. Supply.

base layer; MDC is similar, especially after CoolConferencing enforces substream priority. With Partitioning, for a single source, only receivers requesting the same version can share content with each others; as a result, the upload bandwidth utilization of Partitioning could be lower in certain scenarios. This observation is consistent with other researches [29].

An interesting observation is that: when considering coding efficiency, the results are different. We consider the coding overhead in the above results by multiplying an effective factor (*i.e.*, 0.6 for MDC and 0.7 for SVC). As shown in Figure 11, even with lower bandwidth utilization, Partitioning sometimes has superior performance than MDC/SVC when the supply ratio is equal to or higher than 1. The insight is that we might need to use different multi-rate support schemes under different supply ratios.

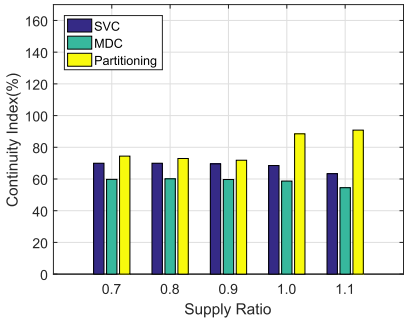


FIGURE 11. Throughput when considering coding efficiency.

C. PERFORMANCE COMPARISON WITH OTHER VIDEO CONFERENCING SYSTEMS

In this subsection, we provide numerical results to evaluate the performance of the CoolConferencing compared to other video conferencing systems and validate the advantage of the proposed scheme.

Specifically, we compare the performance of the proposed CoolConferencing with the layered distribution-based P2P video conferencing proposed in [29], as well as the single-rate scheme called Celerity [27]. In the evaluation the number of peers participating the video conferencing is set to be twelve and the network topology is set to be the same as in Fig. 6. In particular, for CoolConferencing, performances with or without Helper are both evaluated. It is noted that the video conferencing scheme in [29] employs SVC as well, which considers the bottleneck of both the uplink and downlink bandwidth. It is essentially a Mutualcast scheme, which employs 1-hop and 2-hop Mutualcast trees to distribute the layered encoded video of each source. On the other hand, Celerity can handle both bandwidth and delay restrictions which is superior to the Mutualcast scheme without considering the delay effect.

From Fig. 12, we can observe that the proposed CoolConferencing outperforms the Mutualcast system in [29] as well as the Celerity in [27] in the whole region of supply rate. And as the supply ratio decreases, the gains over

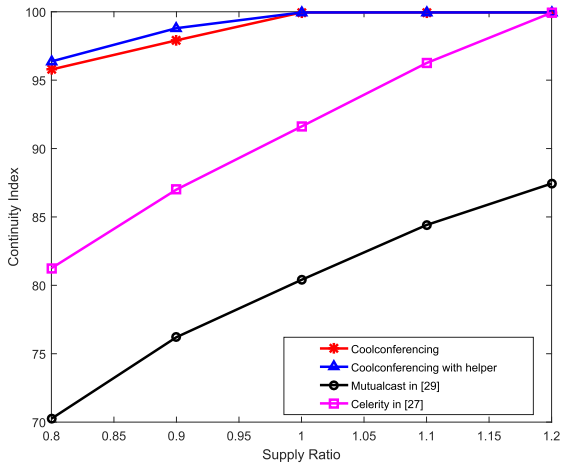


FIGURE 12. Comparison with Mutualcast in [29] and Celerity in [27].

Mutualcast and Celerity become more and more significant. Specifically, when the supply ratio of the system equals to 1, the CoolConferencing can achieve around 25% gain over Mutualcast [29]. The reason is that, packet loss and transmission delay is ignored in the design of the system in [29], while CoolConferencing takes uplink and downlink channel piece missing ratio as well as the nodes' delay map information into consideration explicitly, and hence is able to adapt to the realistic network bottleneck. On the other hand, both CoolConferencing and Celerity incorporate the delay into the scheme design, and both have better performance against Mutualcast. In particular, CoolConferencing is superior to Celerity and when the supply ratio of the system equals to 1, the CoolConferencing can achieve around 9% gain over Celerity [27]. This is because CoolConferencing can support multi-rate and when the supply ratio is small, CoolConferencing can employ multi-rate backoff and use only one layer to transmit. Note that the supply ratio increases, the continuity index of them will reach unanimity gradually. Moreover, it is shown that when a Helper is enabled in CoolConferencing, the continuity of CoolConferencing is enchanced in the whole supply ratio regime, and the gain over Mutualcast [29] becomes even more manifest.

VI. CONCLUSION

This paper proposes a P2P-based any-view video conferencing system CoolConferencing with the modularized designs, which is a robust system that meets all four requirements simultaneously. In the design, we consider the robust and flexible framework which can support more free-perspective selection in video conferencing scenario. The core operations follow the easy-to-implement, robust and resilient data-driven principle, and can adapt to network dynamic distributedly and quickly. Different from other previous works, we fully take account of the networking issues, heterogeneous clients, and scheduling algorithm. Extensive evaluation to demonstrate that CoolConferencing could deliver close to optimum performance under various network environments.

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for their valuable comments.

REFERENCES

[1] Y. Xu, C. Yu, J. Li, and Y. Liu, "Video telephony for end-consumers: Measurement study of Google+, iChat, and Skype," in *Proc. ACM Conf. Internet Meas. Conf.*, 2012, pp. 371–384.

[2] Y. Lu, Y. Zhao, F. Kuipers, and P. Van Mieghem, "Measurement study of multi-party video conferencing," in *Proc. 9th Int. IFIP TC 6 Netw. Conf.*, Chennai, India, May 2010, pp. 96–108.

[3] L. De Cicco, S. Mascolo, and V. Palmisano, "Skype video congestion control: An experimental investigation," *Comput. Netw.*, vol. 55, no. 3, pp. 558–571, 2011.

[4] X. Zhang, Y. Xu, H. Hu, Y. Liu, Z. Guo, and Y. Wang, "Profiling Skype video calls: Rate control and video quality," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 621–629.

[5] X. Zhang, Y. Xu, H. Hu, Y. Liu, Z. Guo, and Y. Wang, "Modeling and analysis of Skype video calls: Rate control and video quality," *IEEE Trans. Multimedia*, vol. 15, no. 6, pp. 1446–1457, Oct. 2013.

[6] *Skype*. Accessed: Sep. 2017. [Online]. Available: http://www.skype.com

[7] *Google+*. Accessed: Sep. 2017. [Online]. Available: http://plus.google.com

[8] *iChat*. Accessed: Sep. 2017. [Online]. Available: http://www.apple.com/macosx/apps/all.html#ichat

[9] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth multicast in cooperative environments," *ACM SIGOPS Oper. Syst. Rev.*, vol. 37, no. 5, pp. 298–313, 2003.

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

[11] C. Tian, R. Alimi, Y. R. Yang, and D. Zhang, "ShadowStream: Performance evaluation as a capability in production Internet live streaming networks," in *Proc. ACM SIGCOMM 2012 Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2012, pp. 347–358.

[12] Y. Huang, T. Z. J. Fu, D.-M. Chiu, J. C S. Lui, and C. Huang, "Challenges, design and analysis of a large-scale p2p-VoD system," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 375–388, Oct. 2008.

[13] H. H. Liu, Y. Wang, Y. R. Yang, H. Wang, and C. Tian, "Optimizing cost and performance for content multihoming," in *Proc. ACM SIGCOMM Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2012, pp. 371–382.

[14] G. Zhang, W. Liu, X. Hei, and W. Cheng, "Unreeling Xunlei Kankan: Understanding hybrid CDN-P2P video-on-demand streaming," *IEEE Trans. Multimedia*, vol. 17, no. 2, pp. 229–242, Feb. 2015.

[15] J. Jansen, P. Cesar, D. C. A. Bulterman, T. Stevens, I. Kegel, and J. Issing, "Enabling composition-based video-conferencing for the home," *IEEE Trans. Multimedia*, vol. 13, no. 5, pp. 869–881, Oct. 2011.

[16] A. Akella, S. Seshan, and A. Shaikh, "An empirical evaluation of wide-area Internet bottlenecks," in *Proc. 3rd ACM SIGCOMM Conf. Internet Meas.*, 2003, pp. 101–114.

[17] N. Hu, L. Li, Z. M. Mao, P. Steenkiste, and J. Wang, "Locating Internet bottlenecks: Algorithms, measurements, and implications," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 41–54, Oct. 2004.

[18] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically-aware overlay construction and server selection," in *Proc. 21st Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM)*, vol. 3. Jun. 2002, pp. 1190–1199.

[19] M. Chen, M. Ponec, S. Sengupta, J. Li, and P. A. Chou, "Utility maximization in peer-to-peer systems," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 36, no. 1, pp. 169–180, Jun. 2008.

[20] H. Zhang, J. Wang, M. Chen, and K. Ramchandran, "Scaling peer-to-peer video-on-demand systems using helpers," in *Proc. 16th IEEE Int. Conf. Image Process. (ICIP)*, Nov. 2009, pp. 3053–3056.

[21] Y. He and L. Guan, "Improving the streaming capacity in P2P VoD systems with helpers," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jun./Jul. 2009, pp. 790–793.

[22] C. F. Lai, H. Wang, H. C. Chao, and G. Nan, "A network and device aware QoS approach for cloud-based mobile streaming," *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 747–757, Jun. 2013.

[23] X. Wang, M. Chen, T. T. Kwon, L. T. Yang, and V. C. M. Leung, "AMES-cloud: A framework of adaptive mobile video streaming and efficient social video sharing in the clouds," *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 811–820, Jun. 2013.

[24] M. Yang, T. Groves, N. Zheng, and P. Cosman, "Iterative pricing-based rate allocation for video streams with fluctuating bandwidth availability," *IEEE Trans. Multimedia*, vol. 16, no. 7, pp. 1849–1862, Nov. 2014.

[25] M. Ponec, S. Sengupta, M. Chen, J. Li, and P. A. Chou, "Multi-rate peer-to-peer video conferencing: A distributed approach using scalable coding," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jun./Jul. 2009, pp. 1406–1413.

[26] M. Ponec, S. Sengupta, M. Chen, J. Li, and P. A. Chou, "Optimizing multi-rate peer-to-peer video conferencing applications," *IEEE Trans. Multimedia*, vol. 13, no. 5, pp. 856–868, Oct. 2011.

[27] X. Chen, M. Chen, B. Li, Y. Zhao, Y. Wu, and J. Li, "Celerity: A low-delay multi-party conferencing solution," in *Proc. 19th ACM Int. Conf. Multimedia*, 2011, pp. 493–502.

[28] Y. Zhao, Y. Liu, C. Chen, and J. Zhang, "Enabling P2P one-view multiparty video conferencing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 73–82, Jan. 2013.

[29] E. Kurdoglu, Y. Liu, and Y. Wang, "Dealing with user heterogeneity in P2P multiparty video conferencing: Layered coding versus receiver partitioning," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr./May 2014, pp. 239–244.

[30] C. Liang, Y. Guo, and Y. Liu, "Investigating the scheduling sensitivity of P2P video streaming: An experimental study," *IEEE Trans. Multimedia*, vol. 11, no. 3, pp. 348–360, Apr. 2009.

[31] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103–1120, Sep. 2007.

[32] Y. Wang, A. R. Reibman, and S. Lin, "Multiple description coding for video delivery," *Proc. IEEE*, vol. 93, no. 1, pp. 57–70, Jan. 2005.

[33] H. Hu, X. Zhu, Y. Wang, R. Pan, J. Zhu, and F. Bonomi, "Proxy-based multi-stream scalable video adaptation over wireless networks using subjective quality and rate models," *IEEE Trans. Multimedia*, vol. 15, no. 7, pp. 1638–1652, Nov. 2013.

[34] E. Magli, M. Wang, P. Frossard, and A. Markopoulou, "Network coding meets multimedia: A review," *IEEE Trans. Multimedia*, vol. 15, no. 5, pp. 1195–1212, Aug. 2013.

[35] B. Li and J. Liu, "Multirate video multicast over the Internet: An overview," *IEEE Netw.*, vol. 17, no. 1, pp. 24–29, Jan. 2003.

[36] T.-Y. Huang, N. Handigol, B. Heller, N. McKeown, and R. Johari, "Confused, timid, and unstable: Picking a video streaming rate is hard," in *Proc. ACM Conf. Internet Meas. Conf.*, 2012, pp. 225–238.

[37] C. Liang, M. Zhao, and Y. Liu, "Optimal bandwidth sharing in multiswarm multiparty P2P video-conferencing systems," *IEEE/ACM Trans. Netw.*, vol. 19, no. 6, pp. 1704–1716, Dec. 2011.

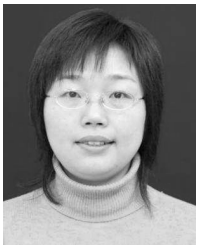
[38] Y. Feng, B. Li, and B. Li, "Airlift: Video conferencing as a cloud service using inter-datacenter networks," in *Proc. 20th IEEE Int. Conf. Netw. Protocols (ICNP)*, Oct./Nov. 2012, pp. 1–11.

[39] Y. Huo, M. El-Hajjar, R. G. Maunder, and L. Hanzo, "Layered wireless video relying on minimum-distortion inter-layer FEC coding," *IEEE Trans. Multimedia*, vol. 16, no. 3, pp. 697–710, Apr. 2014.

[40] M. Alizadeh *et al.*, "Data center TCP (DCTCP)," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, pp. 63–74, Oct. 2011.

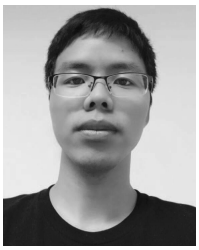
WEIMIN WU received the B.Eng. degree in computer software from Xidian University, Xi'an, China, in 1992, the M.Eng. degree in computer application from Sichuan University, Chengdu, China, in 1995, and the Ph.D. degree in communications and information systems from the Huazhong University of Science and Technology, Wuhan, China, in 2007. He is currently an Associate Professor with the Department of Electronics and Information Engineering, Huazhong University of Science and Technology. His current research interests are in the areas of Internet streaming, broadband wireless communications, and networking.

HANZI MAO received the B.Eng. and M.Eng. degrees in telecommunication engineering from the Huazhong University of Science and Technology. She is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Texas A&M University. Her current research interests include data mining, machine learning, and big data.



YI WANG received the B.S., M.S., and Ph.D. degrees from the Department of Electronics and Information Engineering, Huazhong University of Science and Technology, China, in 2000, 2003, and 2009, respectively. She is currently a Lecturer with the School of Electronics Information and Communications, Huazhong University of Science and Technology. Her research interest is cloud computing.

JI WANG received the B.S. degree from the School of Electronic Information and Communications, Huazhong University of Science and Technology, in 2008, and the Ph.D. degree from the School of Information and Communications Engineering, Beijing University of Posts and Telecommunications, in 2013. He is currently a Post-Doctoral Fellow with the Huazhong University of Science and Technology. His research interests include massive MIMO, energy efficiency, and Internet streaming.



WENKAI WANG received the B.Eng. degree from the School of Electronic Information and Communications, Sichuan University, Chengdu, China. He is currently pursuing the master's degree with the Huazhong University of Science and Technology. His research interests include peer-to-peer networks, broadband wireless communications, and networking.

CHEN TIAN received the B.S., M.S., and Ph.D. degrees from the Department of Electronics and Information Engineering, Huazhong University of Science and Technology, China, in 2000, 2003, and 2008, respectively. From 2012 to 2013, he was a Post-Doctoral Researcher with the Department of Computer Science, Yale University. He was an Associate Professor with the School of Electronics Information and Communications, Huazhong University of Science and Technology. He is currently an Associate Professor with the State Key Laboratory for Novel Software Technology, Nanjing University, China. His research interests include data center networks, network function virtualization, distributed systems, Internet streaming, and urban computing.