PAPER

# Cloud-Assisted Peer-to-Peer Video Streaming with Minimum Latency*

**Satoshi FUJITA**[†a)], ***Member***

**SUMMARY** In this paper, we consider cloud-assisted Peer-to-Peer (P2P) video streaming systems, in which a given video stream is divided into several sub-streams called stripes and those stripes are delivered to all subscribers through different *spanning trees of height two*, with the aid of cloud upload capacity. We call such a low latency delivery of stripes a 2-hop delivery. This paper proves that if the *average* upload capacity of the peers equals to the bit rate of the video stream and the video stream is divided into $a$ stripes, then 2-hop delivery of all stripes to $n$ peers is possible if the upload capacity assisted by the cloud is $3n/a$. If those peers have a uniform upload capacity, then the amount of cloud assistance necessary for the 2-hop delivery reduces to $n/a$.

***key words:*** *P2P video streaming system, cloud assistance, 2-hop delivery scheme*

## 1. Introduction

Video streaming over Peer-to-Peer (P2P) networks has attracted considerable attention in the past two decades [4]–[6], [12], [16]–[19], [21], [22], [24], [25]. The basic idea behind **P2P video streaming** is that peers participating in the network contribute their resources (i.e., upload capacity) to help the continuous dissemination of streaming data. There are various types of P2P video streaming systems, e.g., tree-based systems, mesh-based systems and their hybrid (see Sect. 2 for the details). In tree-based systems [5], [6], [18], [26], peers are organized in a tree-structured overlay and the streaming data which is "fed" by the media server located at the root of the tree, is delivered to the downstream peers by repeating store-and-relay operation. It is known that the weakness of such a tree-structured streaming against churn, faults and attacks could be effectively tolerated by adopting *multiple trees* instead of a single tree [4], [17]; namely by dividing the given stream into $a$ stripes $s_1, s_2, \ldots, s_a$ and by delivering those stripes through different spanning trees. Such a division of a video stream is generally realized in such a way that the $j^{th}$ stripe, for $1 \leq j \leq a$, consists of the $(ai + j)$th chunks in the given stream for $i \geq 0$ [1].

Let us consider a multi-tree-structured P2P consisting of $n + 1$ peers. We assume that any two peers in the P2P system are connected by bi-directional communication links (e.g., through UDP or TCP connections) so that they can directly communicate with each other. For simplicity, we assume that the capacity of each link and the download capacity of each peer through a link are both large enough, while the *upload capacity* of each peer is bounded by a constant. A video stream of unit bit rate is issued by a designated peer called **source**, and is subscribed by the other $n$ peers. Each peer $p$ has upload capacity of amount $u(p)$ so that it can simultaneously upload at most $u(p) \times a$ stripes to the other peers. Each peer can *forward* received stripes to other peers as long as the amount of simultaneous uploads does not exceed $u(p)$.

The delivery of a stripe to peers in the P2P system is said to be $k$**-hop delivery** if *all of n peers* receive the stripe through a path of length at most $k$ [1]. For example, in 1-hop delivery, all peers directly receive the stripe from the source. It is natural to request that $k$-hop delivery must be realized for all stripes for sufficiently small $k$'s such as two or three [4], [23] (see Sect. 2 for the details). In the following, a scheme which realizes a $k$-hop delivery of *all stripes* is simply called a $k$**-hop delivery scheme** [1]. If the upload capacity of each peer is sufficiently large, we can easily realize a $k$-hop delivery scheme for relatively small $k$'s. For example, 1-hop delivery to $n$ peers is possible if (and only if) the upload capacity of the source is $n$ so that it can directly feed the (complete) video stream to all subscribers.

As for the 2-hop delivery of $a$ stripes to $n$ *uniform* peers, the following tight bounds are known [1]. Suppose that $u(p) = u$ for all $p$'s. Then

1. When $a > n$ and $a$ is not a multiple of $n$, 2-hop delivery is possible if and only if $u \geq 1 - \frac{\mu-1}{a}$ where $\mu$ is the smallest integer satisfying the following inequality:
$$\left\lceil \frac{n}{1 + \lfloor \frac{\mu-1}{\beta} \rfloor} \right\rceil - \mu \leq \left\lfloor \frac{a}{n} \right\rfloor + \beta,$$
where $a \equiv \beta \pmod{n}$;
2. When $a = cn$ for some positive integer $c$, 2-hop delivery is possible if and only if $u \geq 1$; and
3. When $a \leq n$, 2-hop delivery is possible if and only if $u \geq \max\{\frac{1}{a}(\frac{n}{\lfloor u/a \rfloor} - 1), 1\}$.

In this paper, we consider a P2P video streaming sys-

Manuscript received June 30, 2018.
Manuscript revised October 1, 2018.
Manuscript publicized November 2, 2018.

[†]The author is with the Department of Information Engineering, Graduate School of Engineering, Hiroshima University, Higashi-Hiroshima-shi, 739–8527 Japan.

*A preliminary version of this paper was presented at APDCM 2018 as "On the Cost of Cloud-Assistance in Tree-Structured P2P Live Streaming," by the same author. This work was partially supported by KAKENHI, the Grant-in-Aid for Scientific Research (B), Grant Number 16H02807.

a) E-mail: fujita@se.hiroshima-u.ac.jp

DOI: 10.1587/transinf.2018EDP7229

Copyright © 2019 The Institute of Electronics, Information and Communication Engineers

tem consisting of $n + 1$ *heterogeneous* peers. As the previous results indicate, the upload capacity of each peer of *amount one* seems to be a crucial factor to enable 2-hop delivery of a video stream of unit bit rate. Thus we make an assumption such that *the average upload capacity of peers equals to one*[†]. In addition, to effectively absorb the heterogeneity of the upload capacity, we further assume that the feed of stripes by the source can be assisted by an external cloud (e.g., cloud storage and cloud content delivery network) with a sufficiently large upload capacity, which is modeled by the *increase of the upload capacity of the source*[††]. For example, when the additional capacity provided by the cloud is $\gamma$, it is modeled by the increase of the upload capacity of the source by $\gamma$. The objective of the current paper is to clarify the cost of cloud assistance in such heterogeneous P2P video streaming systems. The main theorem derived in this paper is stated as follows.

**Theorem 1:** Consider the delivery of a video stream of unit bit rate to $n$ peers in a cloud-assisted P2P system. If the average upload capacity of the peers equals to one and the video stream is divided into $a$ stripes, then 2-hop delivery of the stream is possible if the additional capacity provided by the cloud is $3n/a$.

Note that this is a significant improvement of a naive scheme which incurs a cloud cost of amount $n$ to directly feed $a$ stripes to $n$ peers.

The remainder of this paper is organized as follows. Section 2 overviews related work. Section 3 considers the case in which the upload capacity of each peer is uniformly one. Section 4 extends this result to the case in which the average capacity of the peers equals to one. It also shows the results of experimental evaluations of the proposed method. Finally Sect. 5 concludes the paper with future work. This paper is an extended version of a paper presented at APDCM 2018 [9]. The difference to the conference version is summarized as follows: 1) It fixes bugs existing in the proof of the main theorem. In particular, it completely reorganizes Sect. 4 to improve the quality of presentation and the applicability of the derived results. 2) Experimental evaluations of the proposed method are given, which were not included in the conference version.

## 2. Related Work

Video streaming systems based on the P2P technology are widely used in recent years. Those systems can be classified into several types by the way of delivering video streams to the subscribers, e.g., mesh type such as Bullet [11], PRIME [15], CoolStreaming/DONet [22] and a hybrid of mesh and tree such as mTreebone [20]. Among those systems, we focus on multiple trees as the underlying topology of the overlay network.

The idea of using multiple trees for the delivery of video streams was firstly adopted in SplitStream [4]. SplitStream divides a given video stream into several sub-streams (i.e., stripes) and delivers those sub-streams through different spanning trees to have disjoint sets of internal nodes. In other words, in SplitStream, each peer joins at most one spanning tree as an internal node and joins all of the other spanning trees as a leaf node. Such a construction of the set of spanning trees enables peers to contribute their upload capacity with low cost, which balances the load of all peers participating in the streaming system [2], [4], [7].

Theoretical aspects concerned with multiple-tree-based P2P video streaming have also been studied in recent years. Liu [13] considered the problem of minimizing the broadcast time of each chunk contained in the given stream under the constraint such that each peer can upload at most one chunk at a time, and proposed an algorithm which broadcasts every chunk to $n$ subscribers in $\lceil \log_2 n \rceil$ hops. In this algorithm, any two consecutive chunks are delivered through different binomial trees since in order to enable the delivery of chunks in $\lceil \log_2 n \rceil$ hops, every peer receiving a chunk at time $t$ must continuously upload the chunk until the chunk is received by all subscribers (i.e., note that to complete the broadcast of a chunk to $n$ subscribers in $\lceil \log_2 n \rceil$ steps, the number of subscribers receiving the chunk must *double* in each step). A generalization of the Liu's result to the cases in which each peer can upload at most $k$ chunks at a time, was done in [3] and an extension to the cases in which each peer has constant number of neighbors in the overlay was given in [8]. In addition to the above results, the upper bound on the network capacity of multiple-tree-based P2P is discussed in different contexts; e.g., [14] discussed the network capacity of peer-assisted live streaming systems and [10] considered the problem of constructing multiple trees which maximize the network capacity by considering the topology of the underlying physical network.

Zhao *et al.* analyzed the network capacity of multiple-tree-structured P2Ps in the context of one-view multiparty video conferencing (MPVC, for short) [23]. In one-view MPVC, the delivery of each stream is conducted with the support of helper peers in addition to the publisher and subscribers, and to provide a theoretical bound on the network capacity, they assume that each stream can be divided into sub-streams with *arbitrary fractions*; e.g., a stream with bit-rate $r$ can be divided into sub-streams of bit-rate $\epsilon r$ and $(1 - \epsilon)r$ for arbitrary $\epsilon > 0$, while the length of each delivery path is bounded by two.

In [1], Ando and Fujita introduced the notion of $k$-hop delivery for multiple-tree-structured P2P video streaming. They considered P2P systems consisting of homogeneous peers and focused on the upload capacity of peers which enables the 2-hop delivery of $a$ stripes to $n$ subscribing peers. They derived tight lower bounds on the upload capacity for *any* combination of $a$ and $n$. A part of results concerned with the tight lower bound on the upload capacity of peers

[†]This is a natural requirement as a general P2P video streaming system, since the total amount of downloads can not exceed the total amount of uploads.

[††]In practice, such a cloud assistance is realized by: 1) storing chunks received from the media server to the cloud storage and 2) feeding those chunks to the subscribers through edge servers of cloud CDN.

**Table 1** The value of $\beta$ and $m$ for $a = 16$ and $2 \le n \le 17 (= a + 1)$.

| $n$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $n-1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| $\beta$ | 0 | 0 | 1 | 0 | 1 | 4 | 2 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| $\beta+1$ | 1 | 1 | 2 | 1 | 2 | 5 | 3 | 1 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| $\lceil \frac{n}{\beta+1} \rceil$ | – | – | 2 | – | 3 | 2 | 3 | – | 2 | 2 | 2 | 3 | 4 | 5 | 8 | – |
| $m$ | – | – | – | – | – | 2 | 2 | – | 2 | 4 | – | 3 | 2 | – | – | – |

to enable 2-hop delivery of $a = 16$ stripes is illustrated as follows:

$$\begin{cases} 1 & \text{for } 1 \le n \le 6 \\ 1 + 2/a & \text{for } n = 7 \\ 1 & \text{for } 8 \le n \le 9 \\ 1 + 2/a & \text{for } n = 10 \\ 1 + 4/a & \text{for } 11 \le n \le 12 \\ 1 + 3/a & \text{for } 13 \le n \le 14 \\ 1 + 2/a & \text{for } n = 15 \\ 1 & \text{for } 16 \le n \le 17 \\ (n-1)/a & \text{for } 18 \le n \le 33 \text{ and} \\ 2 & \text{for } 34 \le n \le 35. \end{cases}$$

This indicates that although the tight lower bound *equals to one or slightly greater than one* for $n \le a + 1$, it linearly increases as $n$ increases for $n > a + 1$, and requests each peer to have an ability of forwarding $c$ video streams when the number of peers becomes $c \times a$.

## 3. Uniform Case

In the following, we first consider homogeneous P2Ps with unit upload capacity, and then extend the discussion to heterogeneous P2Ps. More concretely, this section proves the following lemma.

**Lemma 1:** Consider the delivery of a video stream of unit bit rate to $n$ peers in a cloud-assisted P2P system. If the upload capacity of each peer is uniformly one and the video stream is divided into $a$ stripes, then the 2-hop delivery of all stripes is possible with the assistance of the cloud capacity of amount $n/a$.

Recall that the 1-hop delivery of a video stream to $n$ subscribers is possible if and only if the upload capacity assisted by the cloud is $n - 1$. The above lemma implies that we can reduce the cloud cost to $1/a$ by allowing the one hop relay of the stripes, as long as the overhead due to the division of a video stream into stripes can be omitted.

### 3.1 When $a \ge n - 1$

First let us consider the case of $a \ge n - 1$. The case of $a < n - 1$ will be considered in the next subsection. Let $a \equiv \beta \pmod{n-1}$. Table 1 summarizes the value of $\beta$ for each combination of $a = 16$ and $2 \le n \le 17$. In the following, we assume $\beta > 0$, without loss of generality. In the table, it corresponds to the cases of $n = 4, 6, 7, 8$ and $10 \le n \le 16$.

We can realize the 2-hop delivery of $a - \beta$ stripes without cloud assistance in the following manner: As the first hop, the source feeds $a - \beta$ stripes to $n - 1$ peers in such a way that each peer receives $(a-\beta)/(n-1)$ stripes, and as the second hop, each peer forwards the received stripes to the other $n - 1$ peers. Note that each peer including the source consumes the upload capacity of amount $1 - \beta/a$ for the delivery of those $a - \beta$ stripes. To realize the 2-hop delivery of $\beta$ remaining stripes, let us consider a partition of the set of peers into subsets so that:

1. the size of each subset is at most $\beta + 1$ and
2. the number of subsets of size $\beta$ or less is at most one.

Such a partition always exists and generates at least one subset of size $\beta + 1$, since $\beta < n - 1$.

Let $X$ be a subset of size $\beta + 1$. We can realize a 2-hop delivery of $\beta$ stripes to $X$ in the following manner: the source feeds $\beta$ stripes to $X$ in such a way that each peer receives exactly one stripe, which is forwarded to the other peers in $X$ in the next hop. For other subsets of size $\beta$ or $\beta + 1$, the 2-hop delivery of $\beta$ stripes is realized in the same way to $X$ by consuming the cloud capacity of amount $\beta/a$ for each subset. Thus in the following, without loss of generality, we assume that there is a (unique) subset $Y$ of size $m \le \beta - 1$.

The value of $m$ for each combination of $a$ and $n$ is shown in Table 1. Note that $m \le \min\{\beta - 1, n/2 - 1\}$ holds, since we can always take a subset $X$ of size $\beta + 1$. This implies that *the problem of delivering a stripes to n peers has been transformed into the problem of delivering β stripes to* $m < n/2$ *peers*. Note that the cloud cost used for the transformation is at most

$$\frac{n - (\beta + m)}{a} \times \frac{\beta}{\beta + 1} < \frac{n - (\beta + m)}{a}$$

since the delivery of $\beta$ stripes to $X$ is not assisted by the cloud. Thus including the cloud capacity, the source capacity consumed so far is at most

$$\frac{a + n - (\beta + m)}{a}.$$

The above transformation can be applied to the resulting problem in a recursive manner. More concretely, the problem of delivering $\beta$ stripes to $m$ peers can be transformed into the problem of delivering $\beta'$ stripes to $m'$ peers, where $\beta \equiv \beta' \pmod{m-1}$ and $m \equiv m' \pmod{\beta+1}$. The source capacity consumed in the first two transformations is at most

$$\frac{a + n - (\beta + m)}{a} + \frac{\beta + m - (\beta' + m')}{a}$$

**Table 2** The value of $m$ for several combinations of $a = 16$ and $n > 17$.

| $n$ | 100 | 101 | 102 | 103 | 104 | 105 | 106 | 107 | 108 | 109 | 110 | 111 | 112 | 113 | 114 | 115 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $m$ | 15 | - | - | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |

$$= \frac{a + n - (\beta' + m')}{a},$$

which does not exceed $1 + n/a$. Such a recursion is applied until the size of the resulting subset becomes one, and at the deepest level of the recursion, we can complete the 2-hop delivery of $a$ stripes to all peers by simply feeding residual stripes to the unique peer in the subset.

**Example 1:** Consider the problem of delivering $a = 16$ stripes to $n = 11$ peers. Since $a \equiv 6 \pmod{n-1}$, we have $\beta = 6$. To deliver $\beta(= 6)$ stripes to $n$ peers, since $\beta + 1 = 7$, the set of peers is divided into subset $X$ of size 7 and subset $Y$ of size 4. The delivery of $6(= \beta)$ stripes to $X$ is conducted by the source, whereas the delivery to $Y$ is conducted with the assistance of the cloud. Now the problem of delivering 16 stripes to 11 peers is transformed into the problem of delivering 6 stripes to 4 peers. Since $6 \equiv 0 \pmod{3}$, this problem is solved by feeding $2(= \lfloor 6/3 \rfloor)$ stripes to three peers, which is forwarded to the other peers in the second hop.

### 3.2 When $a < n - 1$

When $a < n - 1$, we could apply the same technique in the following manner:

- At first, partition the set of peers into subsets so that: 1) the size of each subset is at most $a+1$ and 2) the number of subsets of size $a$ or less is at most one. Let $X$ be a subset of size $a + 1$ and $Y$ be a subset of size $m \le a - 1$. The transition of value $m$ for several combinations of $a = 16$ and $n > 17$ is shown in Table 2.
- By using the unit upload capacity, the source feeds $a$ stripes to $a$ peers in $X$ which is forwarded to the other peers in $X$.
- For peers not in $X$, the delivery of $a$ stripes is realized with the assistance of the cloud. More concretely, for each subset of size $a + 1$ or $a$, the cloud feeds $a$ stripes to $a$ peers in the subset, which is forwarded to the other peers in the subset in the next hop. Since it does not include the delivery to subsets $X$ and $Y$, the cloud capacity spent for those subsets is at most $\frac{n-(a+1+m)}{a}$.
- The 2-hop delivery of $a$ stripes to $Y$ is realized with the cloud capacity of amount $\frac{a+m}{a}$ by the result of the last subsection.

Thus, in total, the source capacity used for the 2-hop delivery of $a$ stripes to $n$ peers is

$$\frac{n-m-1}{a} + \frac{a+m}{a} < 1 + \frac{n}{a},$$

which completes the proof of the lemma.

## 4. Heterogeneous Case

Next, we consider the case in which the upload capacity of each peer is not uniform but has an average capacity equals to one.

### 4.1 Virtual Peers

Recall that the scheme for homogeneous P2Ps relies on the fact that each peer can directly forward received stripes to other peers so that the unit capacity of each peer is fully utilized. This means that if there is a peer $p$ with $u(p) < 1$, then the role of forwarding stripes should be cooperatively conducted with other peers to have enough surplus capacity. In the following, we call such a group of cooperating peers a **virtual peer**. More precisely, a virtual peer is realized by a collection of **actual peers** with their partial upload capacity so that the sum of partial capacities associated with a virtual peer equals to one, where we omit the overhead required for synchronizing such a cooperation. Note that if the task of forwarding a stripe to other virtual peers is cooperatively conducted by $x$ actual peers, then the (cloud-assisted) source should feed $x - 1$ additional stripes to those actual peers to realize a 2-hop delivery of the stripe. In the following, we consider the problem of simulating $n$ virtual peers executing the original scheme by $n$ actual peers with as small amount of cloud cost as possible.

### 4.2 Key Lemmas

Before proceeding to the proof of Theorem 1, let us consider the following match making problem. Let $G$ be a bipartite graph with vertex set $A \cup B$ and edge set $E$; and let $w$ be a function from $A \cup B$ to $\mathbb{R}$ satisfying the following equality:

$$\sum_{v \in A} w(v) = \sum_{v \in B} w(v), \tag{1}$$

where $\mathbb{R}$ denotes the set of reals. Given edge set $E$, we consider an edge weighting function $w' : E \to \mathbb{R}$. Function $w'$ is said to be **consistent** with $w$ if the following equation holds for all $u \in A \cup B$:

$$\sum_{\{u,v\} \in E} w'(u, v) = w(u). \tag{2}$$

For any function $w$ satisfying Eq. (1), there exists $E$ to have function $w'$ consistent with $w$. In fact, if $E = A \times B$, then we can easily obtain such $w'$ by solving the corresponding maximum flow problem. On the other hand, it is also true that not every $E$ connecting $A$ and $B$ has such $w'$ consistent with $w$ (e.g., consider a perfect matching connecting $A$ and $B$). How can we minimize the cardinality of such $E$ ? The following lemma gives an answer to this question.

**Lemma 2:** For any $w$ satisfying Eq. (1), there is a partition of $B$ into subsets $B^1$ and $B^2$ and an edge set $E \subset A \times B$ to have function $w'$ consistent with $w$ so that: 1) each vertex

in $B^1$ is adjacent with exactly one vertex in $A$ and 2) each vertex in $A$ is adjacent with at most two vertices in $B^2$.

*Proof.* Let $b(q)$ denote the *surplus weight* of vertex $q$ which is initialized as $b(q) := w(q)$ for all $q \in A$. Let $\hat{A}$ be a set of vertices in $A$ with a positive surplus capacity, which is initialized as $\hat{A} := A$. In the following, we will *mark* $p \in B$ to indicate that the set of edges incident with $p$ has been determined with their edge weight. Let $\hat{B}$ denote the set of unmarked vertices in $B$, which is initialized as $\hat{B} := B$.

At first, repeat the following steps while it holds $\min_{p\in\hat{B}} w(p) \leq \max_{q\in\hat{A}} b(q)$ (in the following, we call this loop the first loop of the scheme):

1. Let $q$ be a vertex in $\hat{A}$ with $b(q) = \max_{q\in\hat{A}} b(q)$, and $p$ a vertex in $\hat{B}$ with $w(p) = \min_{p\in\hat{B}} w(p)$.
2. Add $\{p, q\}$ to $E$ with edge weight one and mark $p$.
3. Update the surplus weight of $q$ as $b(q) := b(q) - w(p)$, and remove $q$ from $\hat{A}$ if $b(q) = 0$.

We then repeat the following steps until $\hat{B} = \emptyset$ (in the following, we call it the second loop):

1. Sort $\hat{A}$ in such a way that $b(q_i) \leq b(q_{i+1})$ holds for each $i \geq 1$. Let $\sigma_i \stackrel{\text{def}}{=} \sum_{j=1}^{i} b(q_j)$ for brevity.
2. Let $p$ be a vertex in $\hat{B}$ and let $i^*$ be an index satisfying

$$\sigma_{i^*} \leq w(p) \text{ and } \sigma_{i^*+1} > w(p),$$

   where $q_{i^*+1}$ does not exist if $|\hat{B}| = 1$.
3. For each $q \in \{q_1, q_2, \ldots, q_{i^*}\}$, add $\{p, q\}$ to $E$ with edge weight $b(q)$ and remove $q$ from $\hat{A}$.
4. If $\sigma_{i^*} < 1$ then add $\{p, q_{i^*+1}\}$ to $E$ with weight $1-\sigma_{i^*}$ and update the surplus weight of the vertex as $b(q_{i^*+1}) := b(q_{i^*+1}) + \sigma_{i^*} - 1$.
5. Mark $p$ and go to Step 1.

This procedure establishes edges in $E$ in either of the following two ways:

- By taking a part of the surplus weight of a vertex (Step 2 in the first loop), or
- By taking a part of a collection of surplus weights of vertices (Steps 3 and 4 in the second loop).

In other words, the "update" of the surplus weight takes place in two different manners. In the first type, each update marks exactly one vertex in $B$. On the other hand, the update of the second type is applied to each vertex in $A$ at most once, since a part that is not relevant with an edge in the current iteration is used as the first vertex in the next iteration (note that the updated surplus weight must be the smallest in $\hat{A}$). Hence the lemma follows. Q.E.D.

In the resulting bipartite graph with edge set $E$, vertices in $B^2$ are adjacent with at most $|A| + |B^2|$ vertices in $A$, since for each vertex in $B^2$, there are at most two adjacent vertices which are adjacent with other vertex in $B^2$. Hence we have the following corollary.

**Corollary 1:** For any $w$ satisfying Eq. (1), there is an edge set $E$ of cardinality $|A| + |B|$ to have function $w'$ consistent with $w$.

Let $A$ be the set of $n$ actual peers and $B$ be the set of $n$ virtual peers. Since the capacity of peers satisfies Eq. (1), it satisfies the precondition of Lemma 2, which implies that $n$ actual peers can simulate $n$ virtual peers, in such a way that each virtual peer is simulated by a single actual peer or by a collection of actual peers. Although the cloud cost does not increase for the first case, for the second case, it incurs an additional cloud cost since it would require to send the same stripe to several actual peers simulating a virtual peer. The following lemma indicates that we could bound such an additional cloud cost by the number of actual peers simulating the virtual peer.

**Lemma 3:** Fix a simulation of $n$ virtual peers executing the original scheme by $n$ actual peers, and assume that virtual peer $p^*$ is simulated by $x$ actual peers. If $p^*$ is requested to forward $y$ stripes to other virtual peers in the original scheme, then in addition to $y$ stripes sent to $p^*$ from the source, at most $x$ stripes should be sent to $x$ actual peers simulating $p^*$.

*Proof.* Let $Q$ be the set of $x$ actual peers simulating $p^*$ and suppose that each $q_i \in Q$ contributes the capacity of amount $b(q_i)$ for the simulation. Note that $\sum_i b(q_i) = 1$ holds by definition. Let $S$ be the set of $y$ stripes forwarded by $p^*$, where suppose that $p^*$ uses the capacity of amount $w(s_j)$ for the forwarding of stripe $s_j$ in the original scheme (note that since we are assuming that all stripes have uniform bit-rate, $w(s_j)$ is proportional to the number of receivers of stripe $s_j$). Since $\sum_j w(s_j) \leq 1$ by definition, we may assume $\sum_j w(s_i) = \sum_i b(q_i)$, without loss of generality.

By Lemma 2, we can partition $S$ into two subsets $S^1$ and $S^2$ such that: 1) each stripe in $S^1$ is associated with exactly one actual peer in $Q$ and 2) each actual peer in $Q$ is associated with at most two stripes in $S^2$. This implies that although each stripe in $S^1$ is simply sent to the corresponding actual peer and is forwarded to other virtual peers with no additional cloud cost, each stripe in $S^2$ should be received by several actual peers in $Q$. However, as Corollary 1 claims, since the number of edges in $E$ connecting $S^2$ to $Q$ is at most $|Q| + |S^2| = x + |S^2|$, in addition to $|S^2|$ stripes sent to $p^*$ in the original scheme, the amount of additional stripes to be received by actual peers in $Q$ is at most $x$. Hence the lemma follows. Q.E.D.

Lemma 3 holds for each virtual peer which receives a stripe from the (cloud-assisted) source in the original scheme. The number of actual peers associated with a virtual peer coincides with the number of edges incident with the corresponding vertex in the bipartite graph generated in the proof of Lemma 2. Recall that in the bipartite graph, vertex set $B$ is partitioned into two subsets $B^1$ and $B^2$ so that 1) each vertex in $B^1$ is adjacent with exactly one vertex in $A$, and 2) each vertex in $A$ is adjacent with at most two vertices in $B^2$. Since edges connecting $A$ and $B^1$ does

not contribute to the increase of the cloud cost, we should only count the number of edges incident with vertices in $B^2$. Corollary 1 indicates that the number of such edges is bounded by $|A| + |B^2| \leq 2n$, indicating that the additional cloud cost necessary for the simulation is at most $2n/a$, since each stripe consumes capacity of amount $1/a$. Hence the theorem follows.

## 4.3 Evaluation

### 4.3.1 Basic Policy

This subsection evaluates the performance of the proposed method by experiments. In the theoretical analysis given in this and the last sections, we omit several practical issues such as the overhead due to the division of a video stream into stripes and the overhead caused by the cooperative forwarding of a given stripe by several actual peers, since our goal is to find an assignment of given tasks (i.e., stripes) to resources (i.e., actual peers) without leaving any free space (i.e., the upload capacity of peers is fully utilized). On the other hand, parameters observed in actual P2P systems generally makes such a theoretical problem significantly easy. For example, we could frequently observe a situation in which the capacity of an actual peer (e.g., 512 Kbps) is a multiple of the bit rate of a stripe (e.g., 128 Kbps), but in such a restricted (but practical) situation, it is not difficult to assign stripes to actual peers without leaving any free space.

Based on the above observations, we conduct experimental evaluations under the following policy. The first point is to clarify the tightness of estimations given in the proof of Lemma 2, which plays a key role in applying the lemma to the analysis of the cloud cost in the resulting 2-hop delivery scheme. In particular, the number of virtual peers $|B^2|$ which are not marked during the first loop strongly affects the tightness of the derived bound. In addition, the number of edges incident with a vertex in $B^2$ should also give a significant impact to the overhead, while we omit this factor in the theoretical analysis. In the experiments, we further evaluate the impact of the *reserve force* of actual peers to the performance which is fixed to zero in the theoretical analysis since our goal was to derive the cloud cost to enable 2-hop delivery under such a critical situation.

More concretely, in the following experiments, we assume $|A| = |B| = n$ for simplicity. The vertex weight follows a uniform distribution with range $[1, \max]$, where max indicates the *fineness* of the vertex weight. Note that a small max increases the chance of exhausting the free space in $A$ compared with a large max. In addition, if the vertex weight in $B$ is fixed to a constant and the vertex weight in $A$ follows a uniform distribution so that the average weight in $A$ coincides with the constant weight in $B$, then we have $|B^2| \simeq |B|/2$ which was certified through preliminary experiments.

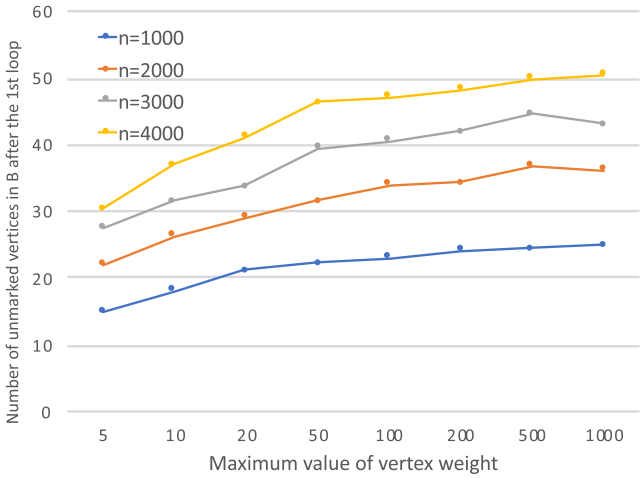


**Fig. 1** The number of vertices in $B$ which are not marked during the first loop of the scheme (averaged over 1000 instances).

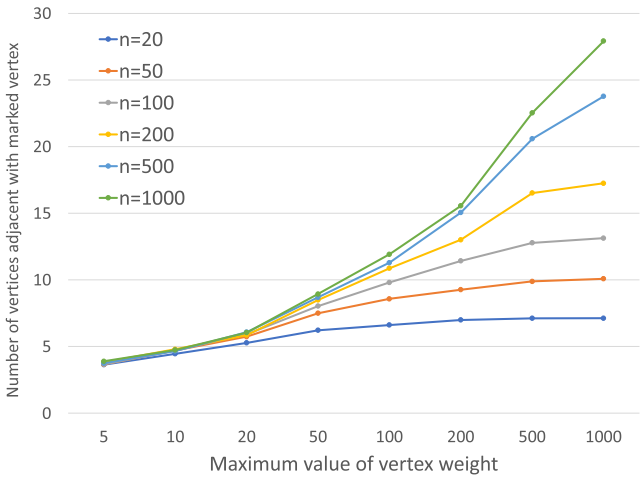


**Fig. 2** The average number of vertices adjacent with a vertex in $B^2$.

### 4.3.2 Results

At first we evaluate the number of vertices $|B^2|$ which are not marked during the first loop. Figure 1 summarizes the results. The horizontal axis is the fineness of the vertex weight, and each point is an average over 1000 instances. We could observe from the figure that although it gradually increases as the fineness of the vertex weight increases, it is bounded by 3% of $n(= |B|)$ even in the finest case, which indicates that almost all vertices in $B$ are associated with exactly one vertex in $A$; namely the assignment of several actual peers to a virtual peer rarely happens.

Next, we evaluate the number of vertices adjacent with a vertex in $B^2$, which corresponds to the number of actual peers simulating the behavior of a virtual peer associated with several peers (while such a case rarely happens as Fig. 1 indicates). Figure 2 summarizes the results. Although it gradually increases as the fineness of the vertex weight increases, it is bounded by a small value even in the finest

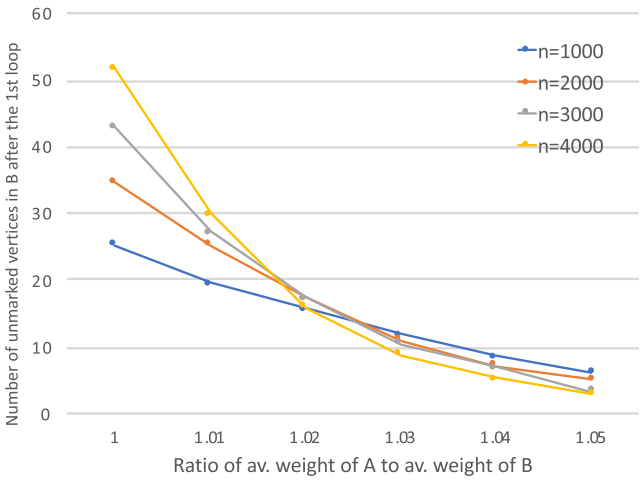


**Fig. 3** The impact of the reserve force of actual peers to the performance.

case: e.g., it is less than 10 when $n = 50$, and it is less than 25 when $n = 500$. This indicates that the proposed method could bound the overhead due to the synchronization of several peers simulating a virtual peer.

Finally, we evaluate the impact of the *reserve force* of actual peers to the performance, in the following manner: 1) while keeping the range of the uniform distribution of the vertex weights in $B$ to $[1, \max]$, we slightly enlarge the range for $A$ to $[1, \rho \max]$ for parameter $\rho \geq 1$, and 2) evaluate $|B^2|$ as was evaluated in Fig. 1. Figure 3 shows the result for $\max = 1000$. The horizontal axis represents $\rho$ varied from 1.00 to 1.05. Although more than 20 vertices are not marked when $\rho = 1.00$, which is consistent with Fig. 1, it rapidly reduces as $\rho$ increases; e.g., more than 99% of the vertices are marked during the first loop when $\rho \geq 1.04$.

## 5. Concluding Remarks

This paper considers the problem of delivering a video stream consisting of several stripes to subscribers in cloud-assisted P2P systems with a minimum latency with as small amount of cloud assistance as possible. We bound the number of hops in the delivery of stripes by two; namely each subscriber must receive each stripe with at most one relay by other subscriber. We prove that if the average upload capacity of the subscribing peers equals to the bit rate of the given video stream and the video stream is divided into $a$ stripes, then we can realize such a low latency delivery of all stripes to $n$ subscribers with the assistance of cloud capacity of amount $3n/a$. This result indicates that the cost of cloud assistance could be arbitrarily small as refining the division of the video stream into stripes.

A future work is to derive a lower bound on the coefficient in the cloud cost.

### References

[1] H. Ando and S. Fujita, "Tight Bounds for Two-Hop Delivery in Homogeneous P2P Video Streaming Systems," Proc. 4th International Symposium on Computing and Networking (CANDAR), pp.1–8, 2016.
[2] B.A. Alghazawy and S. Fujita, "A Scheme for Maximal Resource Utilization in Peer-To-Peer Live Streaming," International Journal of Computer Networks & Communications (IJCNC), vol.7, no.5, pp.13–28, 2015.
[3] G. Bianchi, N.B. Melazzi, L. Bracciale, F.L. Piccolo, and S. Salsano, "Streamline: An Optimal Distribution Algorithm for Peer-to-Peer Real-Time Streaming," IEEE Trans. Parallel and Distributed Systems, vol.21, no.6, pp.857–871, 2010.
[4] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-capacity Multicast in Cooperative Environments," Proc. 19th ACM Symp. Operating Systems Principles (SOSP), pp.298–313, 2003.
[5] M. Castro, P. Druschel, A.-M. Kermarrec, and A.I.T. Rowstron, "SCRIBE: A Large-Scale and Decentralized Application-Level Multicast Infrastructure," IEEE J. Sel. A. Commun., vol.20, no.8, pp.1489–1499, 2006.
[6] Y.-H. Chu, S.G. Rao, and H. Zhang, "A Case for End System Multicast (keynote address)," Proc. ACM Int'l Conf. Measurement and Modeling of Computer Systems (SIGMETRICS), pp.1–12, 2000.
[7] G. Dan, V. Fodor, and I. Chatzidrossos, "On the Performance of Multiple-Tree-Based Peer-to-Peer Live Streaming," Proc. 26th IEEE Int'l Conf. Computer Communications (INFOCOM), pp.2556–2560, 2007.
[8] S. Fujita, "Optimal Serial Broadcast of Successive Chunks," Theoretical Computer Science, vol.575, pp.3–9, 2015.
[9] S. Fujita, "On the Cost of Cloud-Assistance in Tree-Structured P2P Live Streaming," Proc. IPDPS Workshops, pp.820–828, 2018.
[10] X. Jin, W.-P.K. Yiu, S.-H.G. Chan, and Y. Wang, "On Maximizing Tree capacity for Topology-Aware Peer-to-Peer Streaming," IEEE Trans. Multimedia, vol.9, no.8, pp.1580–1592, 2007.
[11] D. Kostić, A. Rodriguez, J. Albrecht, and A. Vahdat, "Bullet: High capacity Data Dissemination using an Overlay Mesh," Proc. 19th ACM Symp. Operating Systems Principles (SOSP), pp.282–297, 2003.
[12] S.-H. Lin, R. Pal, B.-C. Wang, and L. Golubchik, "On Market-Driven Hybrid-P2P Video Streaming," IEEE Trans. Multimedia, vol.19, no.5, pp.984–998, 2017.
[13] Y. Liu, "On the Minimum Delay Peer-to-Peer Video Streaming: How Realtime Can It Be?" Proc. 15th ACM Int'l Conf. Multimedia, pp.127–136, 2007.
[14] S. Liu, R. Zhang-Shen, W. Jiang, J. Rexford, and M. Chiang, "Performance Bounds for Peer-Assisted Live Streaming," Proc. ACM Int'l Conf. Measurement and Modeling of Computer Systems (SIGMETRICS), pp.313–324, 2008.
[15] N. Magharei and R. Rejaie, "PRIME: Peer-to-Peer Receiver-Driven Mesh-Based Streaming," IEEE/ACM Trans. Networking, vol.17, no.4, pp.1052–1065, 2009.
[16] L. Natali and M.L. Merani, "Successfully Mapping DASH Over a P2P Live Streaming Architecture," IEEE Trans. Circuits and Systems for Video Technology, vol.27, no.6, pp.1326–1339, 2017.
[17] V.N. Padmanabhan, H.J. Wang, P.A. Chou, and K. Sripanidkulchai, "Distributing Streaming Media Content using Cooperative Networking," Proc. NOSSDAV'2, pp.177–186, 2002.
[18] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Application-Level Multicast Using Content-Addressable Networks," Proc. NGC'1, vol.2233, pp.14–29, 2001.
[19] H. Shen, Y. Lin, and J. Li, "A Social-Network-Aided Efficient Peer-to-Peer Live Streaming System," IEEE/ACM Trans. Networking, vol.23, no.3, pp.987–1000, 2015.
[20] F. Wang, Y. Xiong, and J. Liu, "mTreebone: A Collaborative Tree-Mesh Overlay Network for Multicast Video Streaming," IEEE Trans. Parallel and Distributed Systems, vol.21, no.3, pp.379–392, 2010.
[21] W. Wu, H. Mao, Y. Wang, J. Wang, W. Wang, and C. Tian, "CoolConferencing: Enabling Robust Peer-to-Peer Multi-Party Video

Conferencing," IEEE Access, vol.5, pp.25474–25486, 2017.
[22] X. Zhang, J. Liu, B. Li, and Y.-S.P. Yum, "CoolStreaming/DONet: A Data-Driven Overlay Network for Peer-to-Peer Live Media Streaming," Proc. 24th Annual Joint Conf. IEEE Computer and Communications Societies, vol.3, pp.2102–2111, 2005.
[23] Y. Zhao, Y. Liu, C. Chen, and J. Zhang, "Enabling P2P One-View Multiparty Video Conferencing," IEEE Trans. Parallel and Distributed Systems, vol.25, no.1, pp.73–82, 2014.
[24] C. Zhao, J. Zhao, X. Lin, and C. Wu, "Capacity of P2P On-Demand Streaming With Simple, Robust, and Decentralized Control," IEEE/ACM Trans. Networking, vol.24, no.5, pp.2607–2620, 2016.
[25] Y. Zhou, T.Z.J. Fu, and D.M. Chiu, "A Unifying Model and Analysis of P2P VoD Replication and Scheduling," IEEE/ACM Trans. Networking, vol.23, no.4, pp.1163–1175, 2015.
[26] S.Q. Zhuang, B.Y. Zhao, A.D. Joseph, R.H. Katz, and J.D. Kubiatowicz, "Bayeux: An Architecture for Scalable and Fault-Tolerant Wide-Area Data Dissemination," Proc. NOSSDAV'1, pp.11–20, 2001.

**Satoshi Fujita** received the B.E. degree in electrical engineering, M.E. degree in systems engineering, and Dr.E. degree in information engineering from Hiroshima University in 1985, 1987, and 1990, respectively. He is a Professor at Graduate School of Engineering, Hiroshima University. His research interests include communication algorithms, parallel algorithms, graph algorithms, and parallel computer systems. He is a member of the Information Processing Society of Japan, SIAM Japan, and IEEE Computer Society.