Peer-Assisted Social Media Streaming with Social Reciprocity

Zhi Wang, *Student Member, IEEE,* Chuan Wu, *Member, IEEE,* Lifeng Sun, *Member, IEEE,* and Shiqiang Yang, *Senior Member, IEEE*

***Abstract*—Online video sharing and social networking are cross-pollinating rapidly in today's Internet: Online social network users are sharing more and more media contents among each other, while online video sharing sites are leveraging social connections among users to promote their videos. An intriguing development as it is, the operational challenge in previous video sharing systems persists, *i.e.*, the large server cost demanded for scaling of the systems. Peer-to-peer video sharing could be a rescue, only if the video viewers' mutual resource contribution has been fully incentivized and efficiently scheduled. Exploring the unique advantages of a social network based video sharing system, we advocate to utilize social reciprocities among peers with social relationships for efficient contribution incentivization and scheduling, so as to enable high-quality video streaming with low server cost. We exploit social reciprocity with two give-and-take ratios at each peer: (1) peer contribution ratio (*PCR*), which evaluates the reciprocity level between a pair of social friends, and (2) system contribution ratio (*SCR*), which records the give-and-take level of the user to and from the entire system. We design efficient peer-to-peer mechanisms for video streaming using the two ratios, where each user optimally decides which other users to seek relay help from and help in relaying video streams, respectively, based on combined evaluations of their social relationship and historical reciprocity levels. Our design achieves effective incentives for resource contribution, load balancing among relay peers, as well as efficient social-aware resource scheduling. We also discuss practical implementation and implement our design in a prototype social media sharing system. Our extensive evaluations based on PlanetLab experiments verify that high-quality large-scale social media sharing can be achieved with conservative server costs.**

***Index Terms*—Social media streaming, social reciprocity, peer incentive, resource allocation.**

I. Introduction

RECENT years have seen the blossom of online social networks (*e.g.*, Facebook [1], Google+ [2]) and video sharing sites (*e.g.*, YouTube [3]), as well as a convergence trend between the two types of systems. More and more media contents (video clips, images, *etc.*) are published and shared among users on social network sites [4], [5], while the video sharing systems are increasingly leveraging social networks to promote their videos and attract viewers [6], [7].

To support efficient video sharing in a social network, similar challenges as faced by traditional online video sharing sites remain: a huge amount of server bandwidth is expected to be provisioned, in order to distribute the large volumes of videos generated and uploaded by users, *e.g.*, more than 60 hours' worth of videos are uploaded every minute in YouTube and served to millions of users per minute. Peer-to-peer (P2P) technology has been advocated to alleviate the server load in video streaming applications [8], [9], such that users (peers) directly send video streams to each other, with less dependence on the dedicated servers. Challenges remain in a peer-assisted design, among which incentivizing sufficient and stable peer bandwidth contribution has been a fundamental one.

There have been a number of P2P incentive designs based on direct or indirect resource trading [10], [11], but none of them has utilized social connections among the peers. The unique setting of a social media sharing system has made very promising a more effective *social reciprocity* based incentive for P2P video streaming over a social network, that exploits the natural intentionality for each peer to help socially connected peers.

In this paper, we design a social media sharing system which utilizes social reciprocity to incentivize effective bandwidth contribution and scheduling at the users, and employ peer-assisted design to distribute video streams with low server cost. We motivate our design with extensive measurement studies of traces collected from Renren [12], one of the largest online social network sites in China. In Renren's traces, we observe that it is common for users to generate videos in the online social network. Meanwhile, the videos generated by users have attracted many viewers. As more and more users in the online social network are generating and consuming videos, it has become a practical problem to design peer-assisted video distribution strategies for social media sharing. In Renren's traces, we also discover that videos are not only shared among friend peers, but also among stranger peers. To design social-aware incentive for peer-assisted video sharing, we consider resource sharing between friends and between non-friends.

An user (peer) in our system is responsible to send its own video stream to all other interested users, either directly if

Manuscript received February 4, 2012; revised July 13, 2012. The associate editors coordinating the review of this paper and approving it for publication were B. Lin, J. Xu, and P. Sinha.

This work has been partially supported by the National Basic Research Program of China (973) under Grant No. 2011CB302206, the National Natural Science Foundation of China under Grant No. 60933013/61272231, the National Significant Science and Technology Projects of China under Grant No. 2012ZX01039001-003, and the Research Grants Council of Hong Kong (RGC GRF Ref: HKU 718710E).

Z. Wang, L. Sun, and S. Yang are with the Department of Computer Science and Technology, Beijing Key Laboratory of Networked Multimedia, Tsinghua University, Beijing, China (e-mail: wangzhi04@mails.tsinghua.edu.cn, {sunlf, yangshq}@tsinghua.edu.cn).

C. Wu is with the Department of Computer Science, the University of Hong Kong, Hong Kong (e-mail: cwu@cs.hku.hk).

Digital Object Identifier 10.1109/TNSM.2012.12.120244

it has sufficient upload bandwidth, or by resorting to *relay* helpers when there are too many viewers. Our key design is to effectively incentivize peers with extra upload bandwidth to serve as relay helpers in others' stream distribution, for which we exploit social reciprocities. We define two light-weighted give-and-take ratios at each peer: (1) peer contribution ratio (*PCR*), which evaluates the historical reciprocity level between a pair of social friends, and (2) system contribution ratio (*SCR*), that records the give-and-take level of the user to and from the entire network. A social reciprocity index (RI) is defined at each user to evaluate each other peer, based on the two give-and-take ratios and the strength of relationship between them two. This index is used in the design of two efficient algorithms for each user to optimally decide which other users to seek relay help from and help in relaying video streams, respectively.

Our design is able to achieve the following effectiveness: (1) peers are maximally incentivized to contribute their available upload resources, (2) load on different relay peers is effectively balanced, and (3) upload bandwidth in the system is efficiently scheduled with social awareness (or social preference), in that users are more inclined to seek help from and help other users with closer social ties.

We have also designed detailed practical protocols to achieve our design, and implemented a prototype social media streaming system. The implementation is extensively evaluated with experiments on PlanetLab, which validate the efficiency and effectiveness of our design in achieving high-quality large-scale social media streaming with low server costs.

This study is based on our previous work on peer-assisted social games [13], but addresses the new problem in social media sharing with social reciprocity. We summarize the major new contributions of this study as follows: (1) We conduct measurement studies based on extensive Renren traces, where the observations well motivate our social reciprocity based incentive design in a peer-assisted social media streaming system; (2) We address the specific design challenge of a peer-assisted social media sharing system, such as how to design algorithms for resource allocation in social media sharing, how to carry out the algorithms when peers who generate the videos are not online, *etc.*; (3) A new prototype social media streaming system is implemented, and our design is extensively evaluated with trace-driven PlanetLab experiments, based on the Renren traces.

The rest of this paper is organized as follows. We discuss related literature in Sec. II, and present our measurement study of Renren in Sec. III. We introduce the system model and the design of two give-and-take ratios in Sec. IV, and present detailed algorithm design in Sec. V. We analyze the effectiveness of our design in Sec. VI. We then present extensive evaluation results based on a prototype implementation and PlanetLab experiments in Sec. VII. Finally, we conclude the paper in Sec. VIII.

II. Related Work

We survey literature on peer-assisted social content sharing and social-aware incentives, respectively.

A. Peer-Assisted Social Content Sharing

The P2P paradigm has been successfully utilized in online file sharing and video streaming, *e.g.*, BitTorrent [14], PPLive [8]. Especially, it has been proven successful in distributing both live [15] and on-demand contents to large numbers of viewers [16]. In P2P streaming, each peer (user or client) caches media chunks of videos it has watched, and distributes these chunks to other peers by direct, mutual exchanges, so as to save the server's upload bandwidth.

As more and more contents are now generated and available in the online social networks, the P2P paradigm has recently been employed to assist in content distribution over an online social network. For P2P file sharing, Pouwelse *et al.* [17] have designed Tribler, which makes use of social connection and trust among users to improve content discovery, recommendation, and downloading. For video sharing, Cheng *et al.* [18] propose a peer-assisted design for distributing short videos on YouTube-like video sharing sites, by exploiting social characteristics such as the "small-world" phenomenon. In Wang *et al.*'s study [19], they verify that online social networks have become important platforms for producing and consuming video contents. They observe that choices of videos a user watches depend not only on the general popularity of the videos, but also on who share the videos. They further design a prefetching strategy to reduce the startup delay in social media streaming, based on users' interest in the videos.

B. Social-Aware Incentives

Incentive engineering has been an important issue of P2P networking since its inception [20]. Different strategies have been designed to encourage peers' mutual sharing of upload resources, *e.g.*, tit-for-tat as used in BitTorrent [21], reputation systems [22], and other game-theoretical approaches [23].

Recently, Liu *et al.* [24] have observed that better file sharing performance can be achieved in some private P2P file sharing systems, by applying an admission control mechanism based on upload/download ratio of each peer. In their design, the system records a balance between the total number of uploaded bytes and the total number of downloaded bytes at each peer. Peers can join the system only when their upload/download ratio is above a certain threshold. Liu *et al.* observe that such admission control can improve the contribution levels of peers in P2P file sharing. In our work, we focus on incentive design in the context of a social media sharing network, and exploit not only system-level give-and-take balances at the peers, but also pair-wise mutual contributions between a pair of socially connected friends.

Liu *et al.* [25] design a network-wide tit-for-tat mechanism for P2P content sharing, where peers can trade resources for contents along a social relation chain. In their design, each pair of peers maintains a limit of the resource that they can provide to each other. Two peers can exchange resources only when a path of social connections can be established between them, and resources and contents will be transferred along the connected social path.

Li *et al.* [26] enable social preferences during message routing in a delay tolerant network, where nodes are more likely to route messages for those with social relationships.

Such a social preference is modeled into an optimal routing decision problem, and the authors design efficient algorithms to solve it accordingly.

III. Motivation from Real-world Traces

To motivate our design of a social media sharing system, we first present our measurement study based on extensive traces collected from Renren.

A. Collection of Traces

Renren is a Facebook-like online social network site, attracting more than 160 million users mainly from China [27]. On Renren, users can create and maintain social connections among each other, as well as post and view various contents generated by connected ones. We obtained runtime traces from the technical team of Renren, which contain about $100,000$ users' online activities in the entire span of July 2010. On Renren, users can import video links to videos hosted on other video sharing sites (*e.g.*, Youku, Tudou, Xunlei), and users clicking on those links can directly view the videos on their Renren pages.

In this paper, our target is to design effective incentive and resource allocation mechanisms for peer-assisted social media sharing. Though Renren is not an exact social media sharing site that we consider (since it does not host video contents itself), the traces nevertheless can reflect patterns of *interest* of users on generating videos and viewing others' videos. Using these traces, we study how peer-assisted mechanisms can be designed to effectively serve videos to users, to satisfy their video viewing interest. We exploit the following information from the traces: (1) The social graph, containing social connections among the users; (2) Video import entry, which records the video links imported by a user from an external video sharing site; (3) Video viewing entry, which records the videos viewed by a user.

B. Importing Videos Is Common in an Online Social Network

The video links shared by Renren users may correspond to videos they themselves generated and uploaded to a video sharing site, or videos produced by others. As an approximate approach, we use the distribution of the number of video links imported by users into Renren, to reflect the distribution of the number of videos that users on a social media sharing site may produce. Fig. 1 illustrates the sorted number of video links imported by users into Renren, based on trace data in the first week of July 2010. We observe that the distribution of the numbers of videos imported by different users is highly skewed. It follows a *zipf* distribution. Correspondingly, the more videos a user in a social media sharing site produces, the more bandwidth it requires to distribute the videos to interested viewers.

C. Videos Imported Can Attract Many Viewers

After the video links are imported into Renren, they attract viewers in the online social network. Fig. 2 plots the number of viewers of a video on Renren, after it is imported into the social network. We observe that some videos can be viewed by many users. In a social media sharing system that we will design, peers whose generated videos attract a large number of views, will need other peers' help in distributing the videos, while other peers whose videos have fewer views can provide such help. In our media sharing design, we model how a single video is distributed by peers, and the model can be extended to address when multiple videos are generated by users.

D. Videos Are Shared Between Friends and Non-Friends

We further investigate the fraction of the views from socially connected friends and that from non-friends. Based on Renren traces, we have observed that around 40% of the video shares are between directly connected friends. Fig. 3 shows the distribution of video views from friends and all, respectively.

In a peer-assisted social media sharing system, videos produced by a peer can be viewed by many other users, which are either socially connected peers of the source, or strangers. When a peer seeks other peers' help in distributing its generated video, there are two cases: if the viewers are mainly from friends, it can rely on the pair-wise reciprocity between itself and socially connected friends; if the viewers are mainly strangers, it can still receive relay help by resorting to its previously cumulated system-wise contribution. Our design in Sec. V will detail the two situations.

IV. System Model

In this section, we present our social media streaming network, define social relationships in the system, and introduce the two give-and-take ratios designed to exploit social reciprocity.

A. Peer-Assisted Media Sharing

We consider a large-scale video sharing network, where a user may generate at most one video, and a number of other users are interested in viewing the video(s). The user that generates a video is referred to as the *source peer* of the video, and users which download and view this video are *viewers* of the video. In our peer-assisted design, the source peer is responsible to distribute its video stream to all the viewers. The peer may directly send the video stream to its viewers if its upload bandwidth suffices, or ask other peers to help relay, when there are too many viewers such that its upload bandwidth is not sufficient to serve all. These helpers are referred to as *relay helpers* or *relays* in short. An illustration of the system is illustrated in Fig. 4, where "S" denotes a source peer that produces the video and "V" represents viewers, which may download the video from the source peer directly or from the relay peer "R". A viewer can further distribute a received video stream to other viewers.

Let $\mathcal{V}_i$ denote the set of viewers of a particular video produced by source peer i. We use $\mathcal{R}_i$ to denote the set of at most K candidate helpers for a source peer i Since a relay peer may help more than one source peer concurrently, we denote the set of source peers requesting relay help from relay j as $\mathcal{S}_j$. In the example scenario in Fig. 5, a, b and c are 3 source peers sharing 3 videos, which we assume have the same streaming rate, respectively. We have viewers $d, f \in \mathcal{V}_a$,

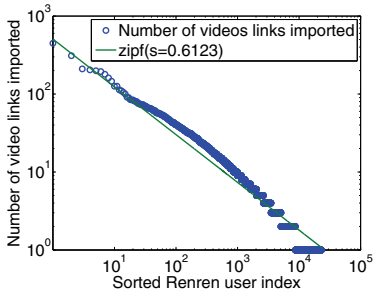


Fig. 1. Number of video links imported by Renren users.

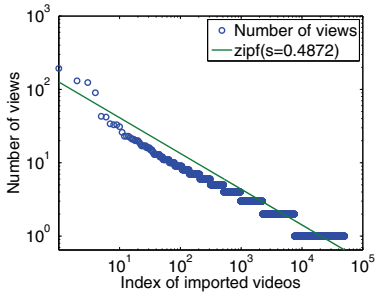


Fig. 2. Number of actual views of imported videos.

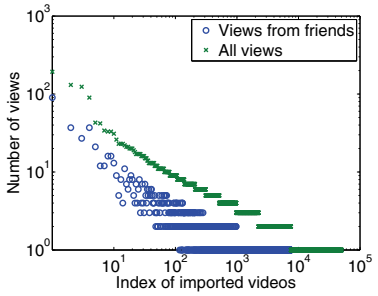


Fig. 3. Number of views from friends and non-friends.

$g \in \mathcal{V}_c$, and $e, h \in \mathcal{V}_b$, relay helpers c, $d \in \mathcal{R}_a$, c, $e \in \mathcal{R}_b$, and $\mathcal{R}_c = \{a, b\}$. Here peer c plays the roles of source peer and relay peer simultaneously.

We assume upload bandwidth at peers constitutes the bandwidth bottleneck, but download bandwidth at peers is always sufficiently large. There are a small number of dedicated servers in the system to serve as backup relay peers, which are resorted to when a source peer cannot find sufficient relay peers. A tracker server keeps track of all online users in the system.

The aim of our design is to effectively incentivize peers to serve as relay helpers for each other using their spare capacity, by exploring social reciprocity among peers, as well as to efficiently schedule upload capacity at peers, which optimizes the utilization of resources in the entire system while taking peers' social preferences into consideration.

B. Social Network Model

We assume that social relationships among the users can be organized into a social graph, where each node represents a peer, and a bidirectional edge exists between two nodes when the two peers are socially connected (*e.g.*, friends, relatives, *etc.*), which we refer to as *social friends* hereinafter. A weight $f_{ij} \in [0, 1]$ is associated with each edge in the social graph, denoting the strength of social connection between peer i and peer j (*e.g.*, strength of friendship). Social relationship is symmetric, *i.e.*, $f_{ij} = f_{ji}$. A larger f_{ij} represents a stronger relationship, and $f_{ij} = 0$ denotes no existing relationship. Viewers of a video produced by peer i can be its social friends or non-friends.

Peers in the video sharing system are supposed to have a *social preference*, *i.e.*, they wish to help their social friends more than the other general population. We exploit *social reciprocity* among peers in our incentive design: (1) We make use of the direct reciprocity between social friends who are socially connected, as peers are naturally willing to help their social friends and receive their help from time to time in return; (2) We exploit indirect reciprocity among a peer and all other peers in the system, where a peer contributes resources to the system expects to receive resource contribution from others as well, although they are not socially connected. Our design will take the above direct and indirect reciprocities into consideration.

C. Two Give-And-Take Ratios

Two ratios are defined to evaluate the level of direct and indirect reciprocities in our system.

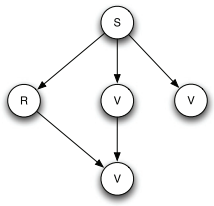


Fig. 4. An illustration of peer-assisted media sharing: a single video.

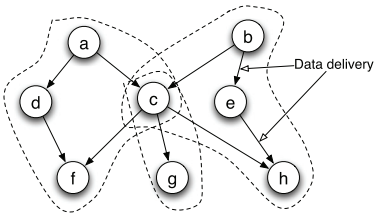


Fig. 5. An illustration of peer-assisted media sharing: multiple videos.

Peer Contribution Ratio (PCR). Peer contribution ratio evaluates the give-and-take balance between two social friends. PCR $W_j(i)$ is defined as the ratio of peer i's upload contribution to peer j over the total mutual contributions between the two:

$$W_j(i) = \frac{C_j(i)}{C_i(j) + C_j(i)}.$$

Here, $C_j(i)$ is the total number of upload bytes that peer i has historically provided for relaying peer j's video stream, and $C_i(j)$ vice versa. $W_j(i) > \frac{1}{2}$ represents that peer i has contributed more to peer j, and $W_j(i) < \frac{1}{2}$ vice versa.

System Contribution Ratio (SCR). To evaluate each peer's contribution in the entire system, we also define a system contribution ratio w_i as follows:

$$w_i = \frac{y_i'}{y_i' + y_i}.$$

Here y_i' is the total number of upload bytes peer i has ever provided for relaying other peers' streams, and y_i is the overall upload bytes of resources others have provided for relaying peer i's stream.

The two ratios are used in our incentive design and resource scheduling, in which peers contributing more to their social friends receive more help from the latter, and for peers with few social friends, providing more relay help to the general others will receive more help in return as well. Detailed design will be discussed in the following section.

In our design, peers who have cached the videos are incentivized to contribute to others. Such candidate peers are utilized in two cases as follows: (1) When the source peer is online, it finds the candidate peers who have cached the videos, and asks them to upload the videos to the viewers; (2) When the source peer is not online, the tracker server is responsible for finding online relays which cache the video and assigning viewers of the video to the relay peers. In both cases, candidate peers who have cached the videos are utilized.

TABLE I
NOTATIONS

Symbol	Definition
PCR	Peer contribution ratio
SCR	System contribution ratio
RI	Social reciprocity index
y'_i	The amount of upload resource peer i has contributed to the system
y_i	The amount of upload resource others have provided to i
$C_i(j)$	The amount of upload resource peer j has provided to i
w_i	The system contribution ratio of peer i
$W_i(j)$	The peer contribution ratio of peer j between the pair of peer i and j
f_{ij}	The social closeness between peer j and peer i
$e_i(j)$	The social reciprocity index of peer j evaluated by peer i
$\mathcal{V}_i$	The viewer set of source peer i
$\mathcal{R}_i$	The relay set of source peer i
$\mathcal{S}_j$	The requesting source set of relay peer j
u_i	The upload capacity of relay peer i
r_i	The stream rate of the video generated by i
$x_{ij}^{(T)}$	The number of viewers that i asks relay j to serve in time slot T
$a_{ji}^{(T)}$	The number of viewers that relay j is to serve for source peer i in time slot T
L	The number of candidate relay peers provided by the tracker server
K	The maximum number of relay peers a source maintains
T_h	The duration of a time slot

The contribution ratios are recorded for the relays accordingly, so that they can be incentivized to help, improving the chance for others to help them in return when they are acting as source peers.

We summarize important notations in the paper in Table I.

V. Detailed Design: Contribution Incentivization and Resource Scheduling

We now present our detailed incentive design for relay resource contribution, through a social reciprocity index that we define, and efficient strategies that peers apply to schedule their resources.

A. Social Reciprocity Index (RI)

We design a social reciprocity index $e_i(j)$ for each peer i to evaluate its perceived contribution level from another peer j, based on the two give-and-take ratios and the strength of social relationship between them two, as follows:

$$e_i(j) = (1 - f_{ij})w_j + f_{ij}W_i(j).$$

The rationale of this index is as follows: If peer j has a stronger social relationship (*i.e.*, large f_{ij}) with peer i, i evaluates j more based on relay help j has provided to itself (*i.e.*, $W_i(j)$). If little social relationship exists between peer i and peer j (*i.e.*, small f_{ij}), i evaluates j more according to relative contributions j has made to the entire system (*i.e.*, w_j).

This index is used in two effective strategies for each source to choose which other peers to seek relay help from, and for each relay to optimally decide which source peers to help, respectively. Specifically, each source will choose peers with smaller RI values it evaluates as relay helpers, and each relay tends to help sources with larger RI values it has evaluated. The idea is four-fold: (1) A source peer i prefers requesting relay help from social friends which it has helped a lot historically (*i.e.*, smaller $W_i(j)$), or from other peers whose contribution level to the entire system is low (*i.e.*, smaller w_j); (2) A relay peer j favors helping its social friends which it has received a lot of help from historically (*i.e.*, larger $W_j(i)$), or other peers whose contribution level to the entire system is high (*i.e.*, larger w_i); (3) Social friends exchange relay help more according to the level of direct reciprocity evaluated by PCRs; (4) Peers with little social connections contribute relay resources among each other more according to SCRs, enabling multilateral reciprocity in the entire system. Detailed algorithms and rationale discussions follow in the next two sub sections.

B. Source Peer's Algorithm: Selecting Relay Helpers

When the number of viewers of the video source peer i produces exceeds its upload capacity, i seeks relay helpers. Three steps are involved: (1) source i chooses relays among a candidate set $\mathcal{R}_i$, which contains potential relay helpers with spare upload bandwidth that i has acquired from the tracker server[1]; (2) source i estimates how many viewers each selected relay peer may possibly help with, and then (3) it assigns specific viewers to each selected relay peer.

1) Selecting Relays: When a source peer asks the tracker server for candidate relays, the tracker will assign it with the relay peers that are geographically closer to its friends. Source peer i ranks all relay candidates in $\mathcal{R}_i$ which have spare upload capacities to share in ascending order of their RIs $e_i(j), \forall j \in \mathcal{R}_i$, and chooses peers to request relay help from in this order. In this way, as discussed in Sec. V-A, source peer i prefers social friends which it has helped a lot and asked little, or other peers which have contributed less to the system but taken more. The reason lies in that those peers are more likely to agree to help the source in serving the video chunks, according to the decision algorithm to be discussed in Sec. V-C, in order to regain their give-and-take balance.

2) Estimating Relay's Available Upload Bandwidth: Each candidate relay peer, which has spare bandwidth (besides sending its own video to its viewers if it is also a source), may potentially help multiple other source peers. To decide how many viewers a relay can upload the video to, i carries out a probing algorithm: For new relay peer j, i randomly decides an initial number of viewers, $x_{ij}^{(0)}$, to relay j. In each following time slot T, if the viewers assigned to relay j can all download the video from j in $T-1$, i will try to assign one more viewer to relay j; otherwise, if only $a_{ji}^{(T-1)}$ viewers ($a_{ji}^{(T-1)} < x_{ij}^{(T-1)}$) are served, i will adjust $x_{ij}^{(T-1)}$ to $a_{ji}^{(T-1)}$,

[1]Implementation details on how a source peer learns about viewers and candidate relay peers will be discussed in Sec. V-D

i.e.,

$$x_{ij}^{(T)} = \begin{cases} x_{ij}^{(T-1)} + 1, & \text{if } a_{ji}^{(T-1)} = x_{ij}^{(T-1)} \\ a_{ji}^{(T-1)}, & \text{if } a_{ji}^{(T-1)} < x_{ij}^{(T-1)} \end{cases}, T = 1, 2, \ldots$$

Since a source peer only maintains at most K relay candidates, the ones with smallest estimate relay capacities will be eliminated from $\mathcal{R}_i$.

3) Assigning Viewers to Relays: After relay peers are selected, the source peer decides which viewers to be served via which relays: we assume that source i is able to estimate the round-trip-times (RTT) between each viewer and each relay (*e.g.*, by referring to a network coordinate system [28]); it assigns a viewer to the relay peer with the smallest RTT in between.

Our design considers only one-hop relay of social media streams, *i.e.*, a relay peer receives the social media stream from the source peer and then forwards to viewers directly, since more relay hops may well add to delay and complexity of the media distribution. In the case that the aggregate upload bandwidth of source i and all its relay peers is not enough to distribute the stream at rate r_i to all viewers, servers are resorted to serve as relays.

The algorithm carried out by each source peer is summarized in Algorithm 1. At source peer i, $RelaySchedule$ is called periodically. In line $1-3$, the stream is delivered to all viewers by source i directly if its upload capacity suffices. Otherwise, it resorts to the relay peers, by selecting the relay helpers that have minimum RIs (line 7), and assigning the number of viewers according to their upload capacity (line $11-12$), which is estimated using the probing strategy discussed. Notice that source i never waits for upload notification from relay peers, since the upload capacities ($x_{ij}(T)$) are updated by upload allocation ($a_{ji}^{(T-1)}$) in the previous round. When the source peer is able to address the remaining viewers, it will stop requesting relay peers (line 9); and source peer i reserves one slot of its upload capacity (equal to its streaming rate r_i), so that it can send its video stream to the server for relaying when capacities from relay peers are not enough.

Algorithm 1 Source Peer's Algorithm.

1: **procedure** RELAY SCHEDULE
2: **if** $u_i \geq |\mathcal{V}_i| r_i$ **then**
3: i will distribute the video stream to all viewers directly
4: **else**
5: Sort $\mathcal{R}_i$ in ascending order of RIs ($e_i(j)$'s)
6: Let y denote the number of viewers to be served by relay peers
7: **for all** relay peer j in sorted list $\mathcal{R}_i$ **do**
8: **if** i is able to distribute to the y viewers **or** its upload bandwidth is smaller than $2r_i$ **then**
9: **break**
10: **end if**
11: $x_{ij}^{(T)} \leftarrow x_{ij}^{(T-1)} + 1$, if $a_{ji}^{(T-1)} = x_{ij}^{(T-1)}$; otherwise $x_{ij}^{(T)} \leftarrow a_{ji}^{(T-1)}$
12: $\min\{x_{ij}^{(T)}, y\}$ viewers are assigned to relay peer j
13: Send to relay j the video stream together with the list of assigned viewers
14: $y \leftarrow y - \min\{x_{ij}^{(T)}, y\}$
15: $u \leftarrow u - r_i$
16: **end for**
17: Send the stream to remaining viewers one by one until i's remaining upload bandwidth is smaller than $2r_i$
18: Resort to the server for relaying to all remaining viewers
19: **end if**
20: **end procedure**

C. Relay Peer's Algorithm: Scheduling Upload Contribution

When a peer has extra service capacity (beyond that used for distributing its own video to its viewers), it may register itself as a candidate relay with the tracker server, and may take itself down from the candidate list when its upload bandwidth is fully used. The voluntary helper registration is incentivized by our upload scheduling algorithm, to be discussed next.

The tracker server may provide each candidate relay peer to multiple source peers, and therefore each candidate relay can receive multiple requests from different sources simultaneously. When relay peer j's spare service capacity is not enough to serve all the source peers' requests, it chooses the source peers to help, prioritizing those with large social reciprocity indices ($e_j(i)$'s) it evaluates. Specifically, a relay periodically decides the source peers it helps in each time slot T, among the set of source peers $\mathcal{S}_j$, which request relay help from itself. On the other hand, once a source peer i is chosen, it is guaranteed that relay j will distribute videos to i's viewers for the duration of the time slot (T_h), in order to avoid inefficiency caused by frequent relay switches.

Let $a_{ji}^{(T)}$ denote the number of viewers that relay j is to serve for source peer i in time slot T. Let $x_{ij}^{(T)}$ be the number of viewers that i asks relay j to serve u_j is the maximum spare upload bandwidth of peer j. The source selection and upload scheduling problem is formulated into the following optimization problem:

$$\max \sum_{i \in \mathcal{S}_j} e_j(i) a_{ji}^{(T)}$$

subject to:

$$a_{ji}^{(T)} \leq x_{ij}^{(T)}, \quad \forall i \in \mathcal{S}_j,$$
$$\sum_{i \in \mathcal{S}_j} a_{ji}^{(T)} r_i \leq u_j,$$
$$a_{ji}^{(T)} \in \{0, 1, 2, \ldots\}, \quad \forall i \in \mathcal{S}_j.$$

To solve this integer linear program for $a_{ji}^{(T)}$'s, we design the following heuristic: Relay peer j first allocates upload capacity of $|\mathcal{V}_j| r_j$ for its own stream distribution. Then j maximally allocates its spare upload bandwidth ($u_j - |\mathcal{V}_j| r_j$) to source peers ($i$'s) in descending order of their social reciprocity indices ($e_j(i)$'s), according to the number of viewers each source has asked j to forward streams to ($x_{ij}^{(T)}$), until its upload bandwidth becomes insufficient to forward a whole stream. In this way, social friends which have helped j a lot

Algorithm 2 Relay Peer's Algorithm.

1: **procedure** UPLOAD ALLOCATE
2: $u \leftarrow u_j - |\mathcal{V}_j| r_j$
3: Sort source peers in $\mathcal{S}_j$ in descending order of RIs ($e_j(i)$'s)
4: **for all** source peer i in sorted list $\mathcal{S}_j$ **do**
5: $a_{ji}^{(T)} \leftarrow \min\{x_{ij}^{(T)}, \lfloor u/r_i \rfloor\}$
6: $u \leftarrow u - a_{ji}^{(T)}$
7: **end for**
8: Send upload allocation $a_{ji}^{(T)}$ to source peer $i, \forall i \in \mathcal{S}_j$
9: On receiving video requests from i's viewers, j serves videos to corresponding viewers
10: **end procedure**

previously, or others which have contributed significantly to the entire system, will be given higher priority.

The algorithm carried out by each relay peer is summarized in Algorithm 2, which is invoked at the beginning of each time slot. In line 2, the relay determines its spare upload capacity, and then allocates it to source peers according to their RIs. After that, the relay peer sends the upload allocation notification $a_{ji}^{(T)}$ to all source peers in $\mathcal{S}_j$ (line 8), and the latter will adjust their requests to other relay peers as discussed in the source peer's algorithm. Upon receiving video requests from viewers of the corresponding source peers, the relay peer uploads the videos to them (line 9).

We note that source peers are allocated upload slots according to how much help they have provided historically, as captured in $e_j(i)$. This incentivizes a peer with spare upload resource to contribute relay help (by registering with the tracker server), such that when it needs relay help for distributing its own video streams, its social reciprocity indices can rank high at other peers, and it can easily receive relay help in return. More discussions on effectiveness of our design will be given in Sec. VI.

D. Implementation Discussions

We next discuss key implementation issues of our design in practice.

1) Maintenance of Give-And-Take Ratios: The amount of resources a peer i has contributed to/taken from its social friends and the entire system, *i.e.*, $C_j(i)$'s, $C_i(j)$'s, y_i', and y_i, are recorded since peer i registered an account with the system. In our implementation, mutual resource contributions between peer i and peer j ($C_j(i)$ and $C_i(j)$) are maintained at both i and j, while system contributions of each peer i (y_i' and y_i) are maintained by the tracker server, similar to that in the existing private P2P file sharing systems [29]. In particular, after relay j helps source i in serving a_{ji} viewers for a time slot, $C_i(j)$ maintained at both peer i and peer j will be increased by the total number of upload bytes $\Delta = a_{ji}^{(T)} r T_h$, while y_i and y_j' maintained at the tracker server will be increased by Δ as well.

2) Tracker Assistance: A source peer i seeks candidate relay helpers from the tracker. In our design, each peer reports its cache state to the tracker, *i.e.*, which videos are stored in the peer's local storage. Upon receiving the requests for relay peers from a source peer, the tracker randomly chooses at most $L = 20$ relay peers that have spare upload bandwidth to share in the system (*i.e.*, peers that have registered themselves at the tracker to help others) and send the relay list to the source peer.

When a source peer of a video is online, it is responsible to distribute its video, and assigns viewers to relay helpers. When the source peer is offline, the tracker server is responsible for finding online relays which cache the video and assigning viewers of the video to the relay peers. It does so by the same source peer's algorithm as in Algorithm 1, and the contribution ratios of the peers (including the offline source peer) will be updated.

Though the tracker can be a potential bottleneck in many peer-assisted systems, this problem is minor in our system due to the following reasons: Given videos in a social media system are generally short, a video, instead of a chunk, is the unit for caching at peers. Our previous measurement study [19] has shown that a large fraction of videos accessed by users in a social media system are very recently published. All these show that the number of cache items at each peer in our system is limited. Hence, the load for cache updates at the tracker server is much lower than that in a traditional P2P VoD system. In addition, distributed tracking mechanisms can be used in large systems for further alleviating the load, such as DHT (Distributed Hash Table) applied in traditional P2P file sharing [30]. In our previous work, we have also confirmed the performance of distributed tracking in P2P VoD streaming [31].

3) Relay Durations: We have designed that a relay j will serve an assigned viewer for at least the duration of one time slot, *i.e.*, T_h. During each time slot, new relay requests from source peers may arrive from time to time, which will be inserted into set $\mathcal{S}_j$; peer j will decide its new upload bandwidth allocation at the beginning of the next time slot. The choice of T_h renders a tradeoff: small T_h may lead to frequent relay switches for source peers, while large T_h may result in less efficient utilization of relay peers' upload bandwidth, as relay requests from source peers with high priority may not be timely addressed. We will evaluate the effects of different T_h values in our experiments.

4) Relay Peer's Caching Policies: Another limited resource at a relay peer is its storage. In the context of social media sharing, newly uploaded videos can typically attract much more viewers than old ones. Therefore, we apply the following caching policy at the peers: (1) When a peer has enough storage capacity, it stores all the videos it has downloaded; (2) When its storage capacity is exceeded, it starts by removing videos that are stored by other source peers in order of the social reciprocity indices.

Finally, note that although we have introduced "time slots" at peers for execution of source and relay algorithms, our system operates in a fully asynchronous fashion: individual peers carry out the designed protocols periodically, while time slots at different peers do not need to be synchronized at all.

VI. ANALYSIS OF DESIGN EFFECTIVENESS

We analyze our design, and show that it achieves the three objectives listed in the introduction, namely peers' maximal

upload contribution, load balancing among relays, and efficient upload bandwidth scheduling with preference towards close social ties.

A. Incentives for Upload Contribution

In our design, relay peers allocate their upload bandwidth to requesting source peers in descending order of their social reciprocity indices (RIs) that it calculates. The more help source peer i has provided to relay peer j or to the entire system, the larger RI ($e_j(i)$) j will evaluate towards i, and the more likely j will help i in relaying its video. Source peer i does not have information about RIs that relay j evaluates towards other requesting sources, and has no idea whether its own RI is large enough to be selected by j. Therefore, peer i has to always keep its RI at a high level, by maximally serving all its own viewers directly (to prevent decrease of RIs due to seeking relay help), and by contributing spare upload resource whenever there is, in order to enhance its system contribution ratio and peer contribution ratios to others (to boost increase of its RIs). Consequently, our design effectively incentives upload bandwidth contributions at all peers in the social media sharing system. Since some peers may have quite small RIs due to their low upload capacities, it could be difficult for them to obtain enough relay help. A possible approach is that these peers can buy "credits" from the system provider, and pay the server or other peers to relay videos for them.

In our design, a potential problem may happen that users who rarely upload videos may not be effectively incentivized to help as relay peers, since these users do not need to resort to others as relay helpers. On one hand, we argue that this problem is minor as follows: Such users are relatively rare in a social media streaming system, where generating and sharing contents (with friends and others) are the basic activities of users, *i.e.*, most users are regularly uploading contents to build reputations and establish influences [32]. On the other hand, we can further incentivize all viewers to help regardless of whether they have contents to upload themselves or not, by associating contribution ratios of a peer with its downloading performance, *e.g.*, viewers with better contribution ratios will be prioritized in receiving the videos from the relays when bandwidths in the system are scarce. Similar mechanisms have been discussed in other P2P file sharing systems [29].

B. Load Balancing at Relays

In source peer's algorithm in Sec. V-B1, a source peer i chooses relays from all candidates in ascending order of their RIs, $e_i(j) = (1-f_{ij})\frac{y'_j}{y'_j+y_j} + f_{ij}\frac{C_i(j)}{C_j(i)+C_i(j)}, \forall j \in \mathcal{R}_i$. Relays with smaller RIs, *i.e.*, which source i has helped a lot but asked little from, or that have contributed less to the system but taken more from, are more likely to be chosen. If relay j is chosen and forwards i's video, y'_j and $C_i(j)$ will increase, then $e_i(j)$ will increase, and peer j's rank lowers in source i's relay selection. In the next round, relay j may not be selected by source i — other candidate relays with smaller contributions will be chosen — until its contribution level becomes relatively low again. In this way, our scheme enables balanced resource utilization at all peers, by always having source peers use relays with historical lowest contribution levels.

C. Social Preference

Besides historical contributions, social closeness f_{ij} between peers is also considered in evaluating RIs. In relay peer's algorithm in Sec. V-C, a relay peer j decides sources to help according to descending order of their RIs, $e_j(i) = (1-f_{ij})\frac{y'_i}{y'_i+y_i} + f_{ij}\frac{C_j(i)}{C_i(j)+C_j(i)}, \forall i \in \mathcal{S}_j$. A source i with closer social relationship (larger f_{ij}) and more historical contribution to j (larger $\frac{C_j(i)}{C_i(j)}$) is more likely to derive a larger $e_j(i)$ and thus has higher priority to be chosen by relay j. Therefore, mutual relay help between peers with close social ties is facilitated, while upload bandwidth in the entire system is efficiently scheduled.

D. Performance in the Presence of Malicious Peers

In a real-world system with our incentive strategies incorporated, malicious peers are likely to exist, which make faked claims on their upload contribution. In our system, a peer's upload contribution can be verified by the receiver of its upload; therefore, the only way for a peer to fake untruthful contribution is by collusion with some other peers [33], where peers in the collusion make faked claims that they have received uploaded streams from each other. These untruthful claims may increase these peers' system contribution ratios, but not the peer contribution ratios.

We argue that such a collusion attack is relatively rare in a social media streaming system, where videos are mainly shared among small social groups and a peer may mainly ask its social friends for the relay. But in case that a collusion attack does occur, traditional countermeasures can be well applied in our design, including the following: i) a reputation-based approach [34], where contribution claims from a peer A are evaluated by another user B according to how much peer B trusts peer A, *e.g.*, according to their social relationship; ii) a credit-based approach [35], where a virtual credit is received for one unit upload bandwidth contribution, peers' contribution ratios are calculated based on the credits peers have earned, and secure mechanisms have been designed for authenticating the credits.

VII. Performance Evaluation

We evaluate the performance of our design based on video sharing traces from Renren and extensive experiments on PlanetLab.

A. Experimental Settings

We have implemented a prototype peer-assisted social media sharing system in C++ programming language, and deployed it on PlanetLab. 200 PlanetLab nodes are used in our experiments, corresponding to the source peers, relay peers, and viewers. Peers in the system generate and view videos according to patterns we summarized from the Renren traces. (1) We follow the distribution of video links that are imported by users into Renren, to emulate the new video generation pattern by source peers in each time slot in our system; (2) We follow the distribution of videos that are viewed by users on Renren to emulate how viewer peers select the videos to watch

in our system; (3) We also follow the social graph on Renren to calculate the social closeness of friends in our experiments.

In particular, the number of videos generated by a source peer over time follows a *zipf* distribution with the skew parameter $s = 0.6123$ This distribution is normalized and each source peer in our system uses this normalized distribution to decide the probability to generate a new video in a time slot. After the source peer generates a video, the probability that a friend of this peer becomes a viewer of the video, follows a probability of 0.1. A friend peer which viewed the video may further inform the video to a friend of itself, according to a random probability of 0.1. The streaming rate of each video is set to 400 Kbps and the streaming duration is 10 minutes. We limit the upload bandwidth of the peers (download bandwidth is never the bottleneck), such that 20% of the nodes have an upload capacity of 1 Mbps, 60% have upload bandwidth of 2 Mbps, and the rest 20% are 4 Mbps.

Social closeness f_{ij} between peers are set following the social graph from Renren traces. We derive the social closeness f_{ij} between peer i and j in the following intuitive fashion: We suppose peers sharing more common friends have closer relationship than those with fewer common friends, and evaluate f_{ij} as the fraction of common friends of peer i and j, $f_{ij} = \frac{|F_i \cap F_j|}{|F_i \cup F_j|}$, where F_i denotes the set of friends of peer i.

B. Server Bandwidth Saved by P2P

We first investigate how much server bandwidth can be saved by the peer-assisted paradigm. In Fig. 6 and 7, we study the upload contribution by peers and the protocol overhead in our system, in terms of the maximum number of relays each source maintains, K, and the length of a time slot, T_h. The upload contribution by peers is defined as the fraction of video chunks that are uploaded by relay peers to the viewers in the system throughout the duration of the experiment. The overhead in our system includes control messages sent by source peers to discover relays, those used by viewers to ask for relay peers and those used by relay peers to register with the tracker server and allocate its upload bandwidth, *etc.*. We evaluate the control overhead using the ratio of the total number of bytes in control messages over the total number of bytes for video distribution in the system.

In Fig. 6, we can see that if T_h is too large or too small, upload contribution by peers is lower. The reason is that for larger T_h, more relay resource is wasted as being reserved for a source peer which may no longer need it. While for smaller T_h, the relay peer needs to reschedule upload bandwidth allocation frequently. With respect to control overhead, Fig. 7 shows that smaller T_h leads to larger overhead, since more control messages are sent among relay peers and source peers for relay scheduling.

On the other hand, we observe that a larger K leads to higher upload contribution by peers, since source peers can select among a larger set of relay candidates, and their requests are more likely to be fully served. However, larger K also leads to higher control overhead, since more messages are sent to maintain these many relays at each source peer.

Based on these observations, we have used the default values of $T_h = 2$ minutes and $K = 30$ in our other experiments, which achieve a good tradeoff between server load reduction and control overhead.

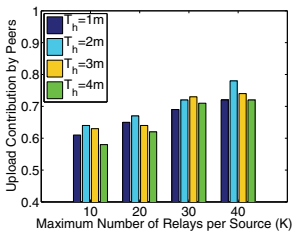


Fig. 6. Server load

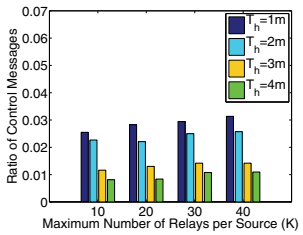


Fig. 7. Protocol overhead

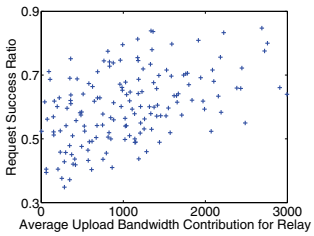


Fig. 8. Request success ratio vs. upload bandwidth contribution for relay at all peers.

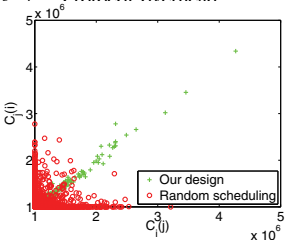


Fig. 9. Mutual contribution between peer pairs.

The above results also show that the bandwidth needed to send control messages for maintaining our P2P system is quite low, as compared to the video distribution bandwidth. In addition, the load on dedicated servers can been significantly alleviated by using our P2P incentive mechanism, to implement a high-quality social media sharing system.

C. Effectiveness of Contribution Incentivization

We investigate the effectiveness of our incentives by evaluating a success ratio of relay requests, issued by different source peers. In our design, the relay request sent from a source peer to a relay may not be served, as relays carry out a source selection algorithm when its spare upload bandwidth cannot serve all the relay requests. If a relay request is not served, the source has to request relay help again from another relay, which introduces additional delay for the viewers to download the videos. Therefore, the request success ratio of a source peer i, defined as the fraction of the number of its relay requests that are fully served by corresponding relays (*i.e.*, at relays $j \in \mathcal{R}_i$ where $a_{ji}^{(T)} = x_{ij}^{(T)}$), over the total number of relay requests it has issued over time, can be used to represent distribution efficiency of a peer's generated videos.

Fig. 8 illustrates the request success ratios of source peers against their respective average upload bandwidth contributed for relaying others' videos over the duration of the experiment. Each sample in the figure represents one peer in the system. We observe that peers with higher upload contribution can obtain larger request success ratio when they need relay help to distribute their own generated videos. In this way, peers are incentivized to contribute more upload bandwidth according to our design.

D. Load Balancing on Peers

We first study mutual relay help between peer pairs. We compare our design with a simple *random scheduling* scheme,

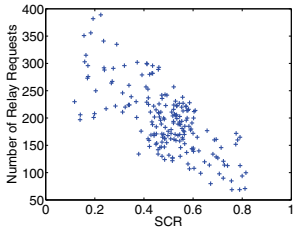


Fig. 10. Relay request load vs. system contribution ratio at all relay peers.

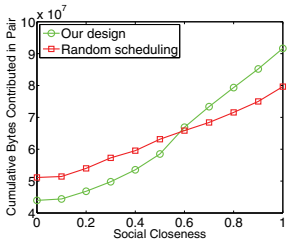


Fig. 11. Cumulative number of packets relayed per source-relay pair over levels of social closeness between the source-relay pair.

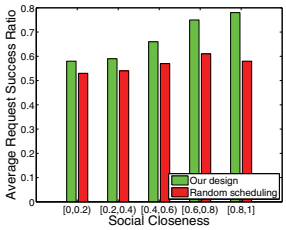


Fig. 12. Average request success ratio vs. social closeness among peer pairs.

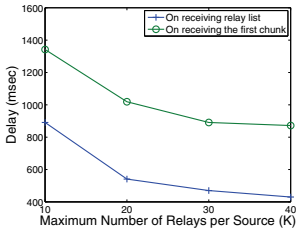


Fig. 13. Startup delays vs. the number of relays at each source.

in which source peers randomly select relays from its candidate pool and relay peers randomly choose sources to help, respectively. Fig. 9 plots the overall number of bytes peer i has helped relay for peer j against the total number of bytes peer j has uploaded for relaying peer i's videos throughout the duration of the experiment, for all pairs of peers in the system. Each sample in the figure represents one peer pair. We observe that mutual resource contribution between two peers is much more balanced with our design than that in the random scheduling scheme.

Fig. 10 illustrates the load of relay requests at each relay peer against its system contribution ratio. The load of relay requests is calculated as the number of requests received by each relay peer in the last 5 minutes, against the system contribution ratio at the beginning of this period. Each sample represents one relay peer. Relay peers with larger SCRs (more contribution to the system) receive less requests than peers with smaller SCRs (less contribution to the system), validating the effectiveness of load balancing among relay peers with our design.

E. Social Preference

Fig. 11 investigates the impact of social closeness between sources and relays on the upload resource allocation at the relays. We categorize source-relay pairs according to the social closeness between the two (*i.e.*, f_{ij}), and count in each category the number of packets each relay has forwarded for the corresponding source throughout the duration of the experiment. This number is further divided by the number of source-relay pairs in each category to generate the number of packets relayed per source-relay pair, and plotted against the social closeness level of the category in a cumulative function fashion in Fig. 11. We can observe that our design achieves better social preference, in that relay help is exchanged more between social friends than that in the random scheduling scheme. The reason lies in that a pair of social friends may provide relay help to each other for many times throughout the video sharing, according to the source and relay selection strategies in our design.

To further illustrate social preference achieved by our design, we divide all pairs of peers into 5 groups according to their social closeness range, and calculate the average request success ratio (defined in Sec. VII-C) between peer pairs in each group, and plot the results in Fig. 12. The results again show that our design achieves better social preference than the random scheduling scheme, in that the average request success ratio is larger when social ties between peer pairs are stronger with our design, while it does not change much with the variation of social closeness levels in the random scheduling scheme.

F. Video Streaming Quality

Finally, we investigate the video streaming quality in our system. In peer-assisted social media sharing, startup delay is a most important factor that decides the streaming quality experienced by viewers. The startup delay consists of two parts: (1) the delay incurred when a viewer discovers relay peers from the source peer, and (2) the time used by the viewer to receive the first video chunk from a relay. Fig. 13 illustrates the two types of delays in our system versus the number of relay peers maintained by the source peer. We observe that smaller K leads to larger delays, and when K is larger than 30, the delays can be bounded. The reason is that when a source peer maintains more relay candidates, it becomes easier for the source peer to assign the requesting viewers with suitable relays to download video chunks from.

VIII. Concluding Remarks

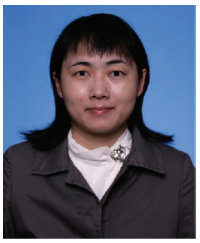
This paper advocates to utilize social reciprocities among peers for efficient contribution incentivization and upload scheduling, to enable efficient social media sharing with low server costs. We exploit social reciprocity with two light-weighted give-and-take ratios at each peer, which record peer's contributions to social friends and to the entire system, respectively. We also design efficient peer-to-peer mechanisms for social media distribution based on a combination of peers' social relationship and historical contribution levels. Through analysis and extensive experiments using a prototype implementation on PlanetLab, we show that our design is able to achieve effective incentives for resource contribution, load balancing among relay peers, as well as efficient social-aware resource scheduling. All these verify that high-quality large-scale social media sharing can be achieved based on our design.

References

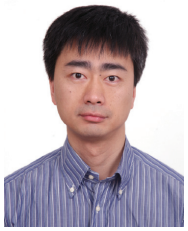
[1] "http://www.facebook.com."
[2] "http://plus.google.com."
[3] "http://www.youtube.com."
[4] H. Kwak, C. Lee, H. Park, and S. Moon, "What is Twitter, a social network or a news media?" in *Proc. 2010 ACM WWW*.

[5] K. Lai and D. Wang, "Towards understanding the external links of video sharing sites: measurement and analysis," in *Proc. 2010 ACM NOSSDAV*.
[6] "http://www.rboke.com/net/youku/2011/0714/7295.html."
[7] N. Savage, "Twitter as medium and message," *Commun. of the ACM*, vol. 54, no. 3, pp. 18–20, 2011.
[8] "http://www.pplive.com."
[9] "http://www.uusee.com."
[10] M. Feldman and J. Chuang, "Overcoming free-riding behavior in peer-to-peer systems," *ACM SIGecom Exchanges*, vol. 5, no. 4, pp. 41–50, 2005.
[11] L. Jian and J. MacKie-Mason, "Why share in peer-to-peer networks?" in *Proc. 2008 International Conference on Electronic Commerce*.
[12] *Renren*, http://www.renren.com.
[13] Z. Wang, C. Wu, L. Sun, and S. Yang, "Peer-assisted online games with social reciprocity," in *Proc. 2011 ACM/IEEE IWQoS*.
[14] "Bittorrent. http://www.bittorrent.com/."
[15] S. Xie, B. Li, G. Keung, and X. Zhang, "Coolstreaming: design, theory, and practice," *IEEE Trans. Multimedia*, vol. 9, no. 8, pp. 1661–1671, 2007.
[16] Y. Huang, T. Fu, D. Chiu, J. Lui, and C. Huang, "Challenges, design and analysis of a large-scale P2P-VoD system," in *Proc. 2008 ACM SIGCOMM*.
[17] J. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D. Epema, M. Reinders, M. Van Steen, and H. Sips, "Tribler: a social-based peer-to-peer system," *Concurrency and Computation: Practice and Experience*, vol. 20, no. 2, pp. 127–138, 2008.
[18] X. Cheng and J. Liu, "Nettube: exploring social networks for peer-to-peer short video sharing," in *Proc. 2009 IEEE INFOCOM*.
[19] Z. Wang, L. Sun, X. Chen, W. Zhu, J. Liu, M. Chen, and S. Yang, "Propagation-based social-aware replication for social video contents," in *Proc. 2012 ACM Multimedia*.
[20] M. Zghaibeh and K. Anagnostakis, "On the impact of P2P incentive mechanisms on user behavior," *2007 NetEcon+ IBC*.
[21] B. Cohen, "Incentives build robustness in BitTorrent," in *Proc. 2003 Workshop on Economics of Peer-to-Peer Systems*.
[22] S. Marti and H. Garcia-Molina, "Taxonomy of trust: categorizing P2P reputation systems," *Computer Networks*, vol. 50, no. 4, pp. 472–484, 2006.
[23] R. Ma, S. Lee, J. Lui, and D. Yau, "A game theoretic approach to provide incentive and service differentiation in P2P networks," in *Proc. 2004 ACM SIGMETRICS*.
[24] Z. Liu, P. Dhungel, D. Wu, C. Zhang, and K. Ross, "Understanding and improving ratio incentives in private communities," in *Proc. 2010 IEEE ICDCS*.
[25] Z. Liu, H. Hu, Y. Liu, K. Ross, Y. Wang, and M. Mobius, "P2P trading in social networks: the value of staying connected," in *Proc. 2010 IEEE INFOCOM*.
[26] Q. Li, S. Zhu, and G. Cao, "Routing in socially selfish delay tolerant networks," in *Proc. 2010 IEEE INFOCOM*.
[27] "http://www.ft.com/cms/s/2/c26ed156-3d23-11e0-bbff-00144feabdc0.html."
[28] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: a decentralized network coordinate system," in *Proc. 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*.
[29] C. Zhang, P. Dhungel, Z. Liu, and K. Ross, "Bittorrent darknets," in *Proc. 2010 IEEE INFOCOM*.
[30] P. Maymounkov and D. Mazieres, "Kademlia: a peer-to-peer information system based on the XOR metric," *Peer-to-Peer Systems*, pp. 53–65, 2002.
[31] Z. Wang, C. Wu, L. Sun, and S. Yang, "Strategies of collaboration in multi-channel P2P VoD streaming," in *Proc. 2010 IEEE GLOBECOM*.
[32] M. Wasko and S. Faraj, "Why should I share? examining social capital and knowledge contribution in electronic networks of practice," *Mis Quarterly*, pp. 35–57, 2005.
[33] M. Feldman, K. Lai, I. Stoica, and J. Chuang, "Robust incentive techniques for peer-to-peer networks," in *Proc. 2004 ACM Conference on Electronic Commerce*.
[34] H. Zhang, A. Goel, R. Govindan, K. Mason, and B. Van Roy, "Making eigenvector-based reputation systems robust to collusion," *Algorithms and Models for the Web-Graph*, pp. 92–104, 2004.
[35] R. Jurca and B. Faltings, "Robust incentive-compatible feedback payments," *Agent-Mediated Electronic Commerce. Automated Negotiation and Strategy Design for Electronic Markets*, pp. 204–218, 2007.

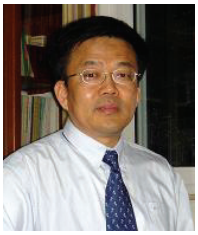
Zhi Wang received the B.E. degree in Computer Science in 2008 from Tsinghua University, Beijing, China. He is currently pursuing the Ph.D. degree in the Department of Computer Science and Technology, Tsinghua University. His research areas include online social network, cloud computing and large-scale multimedia systems. He is a student member of IEEE.



Chuan Wu received her B.E. and M.E. degrees in 2000 and 2002 from Department of Computer Science and Technology, Tsinghua University, China, and her Ph.D. degree in 2008 from the Department of Electrical and Computer Engineering, University of Toronto, Canada. She is currently an assistant professor in the Department of Computer Science, the University of Hong Kong, China. Her research interests include cloud computing, peer-to-peer networks and online/mobile social network. She is a member of IEEE and ACM.



Lifeng Sun received his B.S. and Ph.D. degrees in System Engineering in 1995 and 2000 from National University of Defense Technology, Changsha, Hunan, China; He was on the Post Doctor research of the Department of Computer Science and Technology at Tsinghua University from 2001 to 2003; He is currently an associate professor of the Department of Computer Science and Technology at Tsinghua University. His research interests lie in the areas of online social network, video streaming, interactive multi-view video, and distributed video coding. He is a member of IEEE and ACM.



Shiqiang Yang received the B.E. and M.E. degrees in Computer Science from Tsinghua University, Beijing, China in 1977 and 1983, respectively. From 1980 to 1992, he worked as an assistant professor at Tsinghua University. He served as the associate professor from 1994 to 1999 and then as the professor since 1999. From 1994 to 2011, he worked as the associate header of the Department of Computer Science and Technology at Tsinghua University. He is currently the President of Multimedia Committee of China Computer Federation, Beijing, China and the co-director of Microsoft-Tsinghua Multimedia Joint Lab, Tsinghua University, Beijing, China. His research interests mainly include multimedia procession, media streaming and online social network. He is a senior member of IEEE.