Prefetching Strategy in Peer-Assisted Social Video Streaming * †

Zhi Wang[1], Lifeng Sun[1], Shiqiang Yang[1], and Wenwu Zhu[2]
[1]Tsinghua National Laboratory for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing, China
[2]Microsoft Research Asia, Beijing, China
{wangzhi04@mails., sunlf@, yangshq@}tsinghua.edu.cn, wenwuzhu@microsoft.com

ABSTRACT

Online social network has emerged as the most popular approach for people to directly access multimedia contents. Among these contents, video sharing is a challenging task due to the demand on a large amount of uplink bandwidth at the dedicated server. We leverage a P2P paradigm to alleviate the server to distribute shared videos. By investigating traces obtained from a popular online social network in China, we observe that users' preferences can be predicted. We design a user preference guided prefetching strategy to reduce video startup delays, enabling smooth playback. Simulation experiments show that our design achieves high prefetch accuracy and short startup delay with conservative storage and bandwidth capacities at peers.

Categories and Subject Descriptors

J.4 [**Computer Applications**]: Social and behavioral sciences; C.2.4 [**Distributed Systems**]: Distributed applications

General Terms

Design, Measurement

Keywords

P2P, Social Network, Prefetching, Streaming

1. INTRODUCTION

As one of the most popular applications on the Internet, online social network allows users to establish and maintain their social connections, and to present themselves by posting and sharing various contents [6]. In this paper, we focus on the video sharing in online social network. We have obtained traces from the administrator of Renren [1], one of the largest online social network sites in China. We observe that videos that are originally hosted on video sharing sites, such as Youku [2] and Youtube [3], are aggregating into Renren. It is clear that online social network has emerged as a new platform where users can access video contents conveniently.

Information hosted by online social network can be used to identify users' interests. Cheng *et al.* [4] have proposed a reputation model based on trust between friends to guide users to browse desirable contents, and Yang *et al.* [9] have proposed a movie recommendation system based on rating information shared among friends. In this paper, we make use of such information hosted in the online social network, to guide peers to prefetch videos to enable high-quality social video streaming.

First, we leverage a P2P (Peer-to-Peer) paradigm to enable video streaming to large-scale users in the online social network with conservative server bandwidth. In P2P streaming, each peer (user or client) is capable to cache media chunks of videos it has watched, and distribute these chunks to other peers, saving the server uplink bandwidth. P2P has been proven successful in distributing both live [8] and VoD (Video-on-Demand) contents [7]. Cheng *et al.* [5] also proposed peer-assisted distribution of short videos on video sharing sites like Youtube.

Second, we design a prefetching strategy to enable high-quality streaming. By prefetching the prefix chunks of a video that is likely to be viewed by a user, a short startup delay and smooth playback can be achieved. Our prefetching strategy is based on users' *preferences* gradually learnt from their historical video access patterns. In the online social network, users' preferences of videos are predicted by (1) users' historical selections of *source* users who share the videos, (2) social closenesses between the *viewing users* and the source users, and (3) popularities of the videos. For a user, the un-watched videos are ranked according to the user's preferences of them, such that videos with higher ranks are more likely to be viewed by the user. After that, the user (peer) downloads prefix chunks of the ranked videos in order. The startup delays thus can be reduced when the user plays videos that have been prefetched.

The remainder of this paper is organized as follows. We investigate traces of Renren in Sec. 2. We present our prefetching design in Sec. 3. In Sec. 4, we give the evaluation results. And we conclude the paper in Sec. 5.

*Area chair: Surender Chandra

†Supported in part by National High-Tech Research and Development Plan of China (863) under Grant No. 2009AA01Z328, the National Natural Science Foundation of China under Grant No. 60833009 and No. 60773158, and the National Basic Research Program of China (973) under Grant No. 2006CB303103.

MM'11, November 28–December 1, 2011, Scottsdale, Arizona, USA.

Table 1: Trace Item of Content Viewing

Field	Definition
T	The time stamp of this trace item
Viewer	The user who views the content item
Source	The user who posts or shares the content item
Content	The ID and URL of the content item

2. MEASURING VIDEO SHARING/VIEWING ON RENREN

As one of the largest online social network sites in China, Renren has attracted more than 160 million users as of Feb. 2011. The traces we obtained from Renren contain about $100,000$ users, their social connections, and their interactions. From the traces, we try to understand users' preferences of videos shared by others.

2.1 Traces from Renren

On Renren, users are allowed to share video links from other video sharing sites, *e.g.*, Youku, so that users can directly view these videos on their Renren pages. In the traces, about $21,000$ videos are imported from Youku between July 1st and July 31st, 2010. After videos are imported to Renren, how users view them are recorded by the content viewing trace items, each of which contains four fields as illustrated in Table 1.

2.2 Measurement Results

We are interested to know the impact of *social relationship* on users' preferences of videos, *i.e.*, how users select videos shared on Renren. In Renren's traces, we have made the following observations:

Videos Aggregate in the Online Social Network. Many video sharing sites have provided an easy "share-in-a-click" function for users to share videos with their friends in the online social network. We retrieve the original video sharing sites from the trace items. As shown in Fig. 1, videos on Renren are imported from many external popular video sharing sites. It is clear that videos of traditional video sites are *aggregating* in the online social network. On the other hand, the social relationship and users' preferences hosted by the online social network can improve the video recommendation, making users more likely to access videos directly via the social network.

Users' Preferences of Videos shared by Friends. On Renren, a user is allowed to watch not only videos shared by his friends, but also a much larger number of other popular videos shared by strangers, and automatically recommended by the system. However, we observe that more than 12% of videos views are between friends, *i.e.*, the source and viewing users are socially connected. Considering the huge number of strangers to a user on Renren, this observation indicates clear preferences of users to watch videos shared by their friends.

Weak Correlation between Video and Blog Viewings. Since blog and video are two important types of contents on Renren, we wonder if the access patterns of blogs can be used to predict the access patterns of videos. We investigate the correlation between blog viewing and video viewing. In Fig. 2, each sample shows the number of video views versus the number of blog views between a pair of users during a week. We observe that the correlation is quite weak. Thus, in our design, how users view blogs is not used to predict how they view videos.

Predicting Users' Preferences from Historical Selections. Since users' interests can be relatively stable, a pair of users who have similar interests should be likely to view videos shared by each other for a long period. We select trace items spanning two consecutive weeks to verify it. We calculate the number of views between a pair of users for each week. In Fig. 3, each sample illustrates the number of views between a pair of users in a week versus that in the previous week. The relatively strong correlation indicates that users' historical selections can predict users' future preferences.

Next, we make use of the historical video selection and social closeness information to guide the video prefetching, so that startup delays can be reduced in the video streaming.

3. PREFETCH FOR SOCIAL VIDEO STREAMING

As observed in Renren's traces, users' selections of videos depend on not only what the videos are, but also who share the videos with. As a promising approach to reduce video startup delay, prefetching the prefix chunks of videos has to solve a key problem: among the huge number of un-watched videos, which ones are more likely to be viewed by a user?

According to the observations in Sec. 2, we design a prefetching strategy based on users' preferences guided by information from the online social network. We first discuss how a user's preference is predicted, then give the detailed prefetching design.

3.1 Choosing Videos to Prefetch

In the social video streaming, a viewing user has to choose among many videos to prefetch. In our design, the viewing user's video preference is predicted by (1) who share the videos and (2) what the videos are.

3.1.1 Preference of Source Users

We use the following metrics to predict a viewing user's preference of a source user: (1) His historical selection of the source user, which has been observed to have strong correlation with his future selection; (2) Social closeness between the viewing user and the source user, since friends are observed to be more likely to view each other's shared videos.

Historical Selection of the Source User. In the online social network, viewing user i chooses videos shared by multiple source users in set Y_i, which is the set of source users whose shared videos are likely to be viewed by user i. In our design, Y_i contains all i's friends, and strangers whose shared videos have been watched by i before. We define a pair-wise index h_{ij} to evaluate i's historical selection level of source user j as $h_{ij} = \frac{c_{ij}}{\sum_{k \in Y_i} c_{ik}}$, where $j \in Y_i$ and c_{ij} is the number of historical views that peer i watches videos shared by peer j. Large h_{ij} indicates that user i prefers videos shared by user j, such that user i can still choose videos shared by user j in the future.

Social Closeness between the Viewing User and the Source User. We observe that users are more likely to watch videos shared by their friends on Renren. In our design, such social preference between friends is evaluated by a straightforward metric, which is the fraction of their common friends over their total friends. The social closeness f_{ij} between user i and user j is defined as $f_{ij} = \frac{|F_i \cap F_j|}{|F_i \cup F_j|}$, where $j \in Y_i$ and F_i

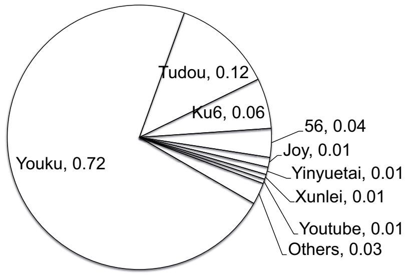

Figure 1: Videos are aggregating in online social network

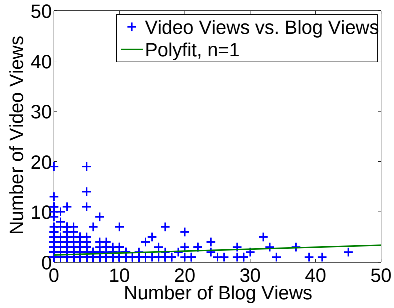

Figure 2: Video views vs. blog views

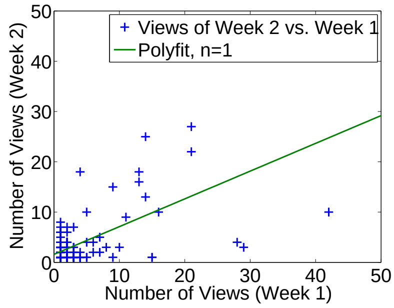

Figure 3: Video preference vs. historical selection

is the set of friends of user i. Intuitively, larger f_{ij} indicates stronger social closeness between i and j.

Preference of a Source User. According to our observations from Renren's traces, both historical selection and social closeness predict a viewing user's future preference of source users. We define a source preference index e_{ij} to evaluate how much viewing user i likes to watch videos shared by source user j as $e_{ij} = w_i f_{ij} + (1 - w_i)h_{ij}$, where w_i is a parameter for a peer to adjust the weights of historical selection and social closeness. Larger w_i indicates that i selects videos more according to friend links. In our design, w_i is dynamically tuned by peer i in each round. At the end of each round, if j' is discovered to be the "best" source user, *i.e.*, the view number of videos shared by j' is the largest, then w_i will be increased by a small amount δ if $f_{ij'} > h_{ij'}$ and $w_i < 1$, or decreased by a δ if $f_{ij'} < h_{ij'}$ and $w_i > 0$. In our experiments, w_i is initially set to 0.5, δ is set to 0.01, and each round is 30 minutes long.

3.1.2 Preference of Videos

Video Popularity. Source preference index e_{ij} evaluates viewer i's preference to choose source user j. However, a source can share multiple videos; while for a viewing user, only being able to select the "best" source users is not enough to do the prefetching. We next give the video rank scheme based on video popularities and source preference indices. The popularity of a video reflects how much viewing users like the video. In our design, the popularity of video k is defined as p_k, which is the number of views that users watch the video.

When a user logs in his account, he is provided a list of un-watched videos shared by his friends and other users. The number of these videos can be very large for a viewing user. We define V_{ji} as the set of source j's shared videos that have not been watched or prefetched by peer i. We give each video $k \in \cup_{j \in Y_i} V_{ji}$ a video preference index $s_i(k)$ as follows:

$$s_i(k) = p_k \sum_{j | k \in V_{ji}} e_{ij}, \quad k \in \cup_{j \in Y_i} V_{ji}.$$

Larger $s_i(k)$ indicates that viewing user i is more likely to watch video k in the future.

3.2 Prefetching The Selected Videos

3.2.1 Downloading The Prefix Chunks

When prefetching a video, a peer actively downloads the prefix chunks of that video, *i.e.*, the first several chunks. The prefetching strategy is carried out by a peer locally when the buffered chunks for the current video can be played for at least 5 seconds (*i.e.*, enough to resist network jitters), or it idles (*i.e.*, no video is being played). The peer keeps a sliding window for the current video, expecting to receive urgent chunks, which are for the next 5 seconds. Only when the sliding window is fully filled, the peer carries out the prefetching. The un-prefetched videos in set $\cup_{j \in Y_i} V_{ji}$ are ranked in $s_i(k)$'s descending order. Videos with larger $s_i(k)$ will be prioritized to be prefetched. When underflow of the sliding window occurs, the peer will suspend the prefetching and focus on downloading chunks of the current video.

Table 2: Bandwidths of Peers [10]

Fraction of Peers	20%	50%	30%
Downlink (kbps)	784	1500	3000
Uplink (kbps)	128	384	1000

3.2.2 Cache Strategy

As a peer's local storage capacity is limited, stale video chunks have to be eliminated to make room for new ones. We divide the whole storage at a peer into 3 regions and give them different priorities as follows: (1) The region for current video (*high priority*): chunks in this region are never eliminated whether they have been played or not, to enable fast seek forward/backward, which are found to be quite normal [11]. (2) The region for prefetching (*normal priority*): we fix the number of maximum chunks for prefetching to K, *i.e.*, a user will stop prefetching when this region is fully filled. In this region, a *FIFO* scheme is used to eliminate stale chunks. (3) The region for other chunks (*low priority*): this region occupies the remained room of the storage, where an *LFU* replacement scheme is used to eliminate the stale chunks. In our design, the dedicated server and peers also prioritize to serve chunks for current video playback over prefetching chunks.

4. PERFORMANCE EVALUATION

We evaluate the performance of our design by simulation experiments driven by Renren's traces.

4.1 Experiment Setup

We choose 500 active users who have viewed the largest numbers of videos. The bitrate and duration of a video are set to 600 Kbps and 10 minutes, respectively. Peers' uplink and downlink capacities are configured in Table 2. The uplink capacity of the dedicated server, who stores all videos,

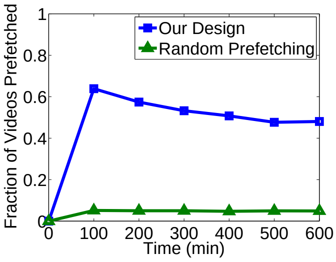

Figure 4: Prefetching accuracy vs. time

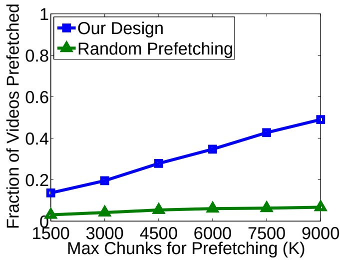

Figure 5: Prefetching accuracy vs. max chunks for prefetching

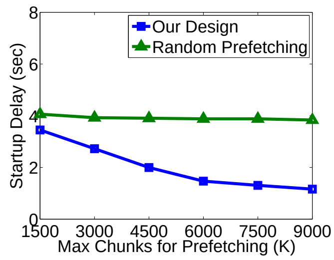

Figure 6: Startup delay vs. max chunks for prefetching

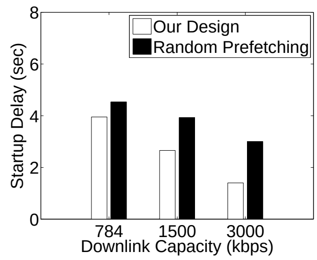

Figure 7: Startup delay vs. downlink capacities at peers

is set to 100 Mbps. The size of each chunk is 1200 bytes. A user chooses the videos to watch as indicated in the traces. A user caches the chunks it has downloaded according to the cache strategy discussed in Sec. 3. Neighboring peers who can help each other are discovered by a tracker server. When downloading chunks, users make use of their neighboring peers, and only download the urgent chunks from the dedicated server to reduce the startup delays.

4.2 Experiment Results

4.2.1 *Prefetching Accuracy*

We first evaluate the prefetching accuracy, which is defined as the fraction of the prefetched videos (at least one of whose prefix chunks has been downloaded *before watching*), over all videos watched by peers. We compare our design with a random prefetching scheme, in which peers download prefix chunks of videos randomly chosen among all videos shared by its friends. Fig. 4 illustrates the prefetching accuracy versus the simulation time. We observe that our design achieves much higher prefetching accuracy than the random scheme. And the prefetching accuracy can reach a stable value after a short period when peers have learnt their interested sources and video popularities.

We also evaluate the impact of the maximum number of chunks for prefetching on the prefetching accuracy. Fig. 5 depicts the average prefetching accuracy versus the maximum number of chunks for prefetching (K). We observe that larger storage capacity leads to higher prefetching accuracy in our design; while storage capacity has little impact on the random prefetching scheme. The reason is that peers have downloaded too many chunks that are never played in the random prefetching.

4.2.2 *Startup Delay*

We also evaluate the startup delays, which are caused by initial buffering when users start videos. Fig. 6 illustrates the average startup delay versus K. From this figure we can see that our design achieves smaller startup delays than the random prefetching scheme. We also observe that larger K leads to shorter startup delays in our design, and has little impact on the random prefetching scheme, mainly due to the much lower prefetching accuracy of the random scheme.

Startup delays are also determined by the downlink capacities at peers. Peers with larger downlink capacities are able to download more chunks, so that more candidate videos can be covered. As illustrated in Fig. 7, we observe that larger downlink capacities in both our design and the random scheme lead to shorter startup delays, and similar downlink capacity achieves much shorter startup delays in our design than in the random prefetching scheme. The reason is that prefetching only works when peers have extra capacities to download prefix chunks of other videos, and our design chooses the most effective chunks for prefetching.

5. CONCLUDING REMARKS

Videos from video sharing sites like Youku are aggregating into online social networks like Renren. In Renren's traces, we observe that users' video viewing preferences can be predicted by their historical selection of source users, social closenesses and the videos' popularities. We design prefetching strategy based on the users' video preferences. Our simulation results verify the effectiveness of our prefetching design, which achieves short startup delays, enabling fast video switching and smooth playback.

6. REFERENCES

[1] http://www.renren.com.

[2] http://www.youku.com.

[3] http://www.youtube.com.

[4] R. Cheng and L. E.K. Bring Order to Online Social Networks. In *Proc. of IEEE INFOCOM*, 2011.

[5] X. Cheng and J. Liu. Nettube: Exploring social networks for peer-to-peer short video sharing. In *Proc. of IEEE INFOCOM*, 2009.

[6] L. Guo, E. Tan, S. Chen, X. Zhang, and Y. Zhao. Analyzing Patterns of User Content Generation in Online Social Networks. In *Proc. of ACM SIGKDD*, 2009.

[7] Y. Huang, T. Fu, D. Chiu, J. Lui, and C. Huang. Challenges, Design and Analysis of A Large-scale P2P-VoD System. In *Proc. of ACM SIGCOMM*, 2008.

[8] S. Xie, B. Li, G. Keung, and X. Zhang. Coolstreaming: Design, Theory, and Practice. *IEEE Transactions on Multimedia*, 9(8):1661–1671, 2007.

[9] X. Yang, G. Y., and L. Y. Bayesian-inference Based Recommendation in Online Social Networks. In *Proc. of IEEE INFOCOM*, 2011.

[10] M. Zhang, Y. Xiong, Q. Zhang, L. Sun, and S. Yang. Optimizing the Throughput of Data-driven Peer-to-Peer Streaming. *IEEE Transactions on Parallel and Distributed Systems*, 20:97–110, 2008.

[11] C. Zheng, G. Shen, and S. Li. Distributed Prefetching Scheme for Random Seek Support in Peer-to-Peer Streaming Applications. In *Proc. of ACM MM*, 2005.