# COOCHING: Cooperative Prefetching Strategy for P2P Video-on-Demand System

Ubaid Abbasi and Toufik Ahmed

CNRS LaBRI Lab. – University of Bordeaux 1
351 Cours de la Libération, Talence Cedex 33405 – France
{abbasi , tad} @ labri. fr

**Abstract.** Most P2P VoD schemes focused on service architectures and overlays optimization without considering segments rarity and the performance of prefetching strategies. As a result, they cannot better support VCR-oriented services. Despite the remarkable popularity in VoD systems, there exists no prior work that studies the performance gap between different prefetching strategies. In this paper we analyze and understand the performance of different prefetching strategies. Our analytical characterization brings us not only a better understanding of several fundamental tradeoffs in prefetching strategies, but also important insights on the design of P2P VoD system. On the basis of this analysis, we finally proposed a cooperative prefetching strategy namely "COOCHING". In this strategy, the segments requested in VCR interactivities are prefetched into session beforehand using the information collected through gossips.

## 1 Introduction

Over the past few years, multimedia communications have become essential part of people's daily life. Multimedia streaming is attracting extensive attention and becomes the most popular activity over the Internet. Video streaming supports a large number of simultaneous users and consumes more network bandwidth as compared to other internet applications [2]. It can be classified into two categories: Live streaming and Video on Demand (VoD). In live streaming systems, the source server broadcast the contents and all the clients play the contents at a same progress. On the other hand, VoD is an interactive multimedia service in which user enjoys the video with completely free choices due to the availability of VCR controls (i.e., forward, backward, resume). The important observation regarding P2P VoD systems is that, users don't watch the video from beginning to end [1]. VoD users performs the seek operations very frequently. This results in increase latency and causes excessive stress on streaming server.

Data Prefetching has been proposed as a technique for reducing the access latency. In this technique, peers prefetch and store various portions of the streaming media ahead of their playing position as shown in Fig. 1. Although it requires the additional bandwidth and storage for prefetching, considering the increasing bandwidth on network and storage capability on local peers, it actually offers a more desirable tradeoff between quality and cost. An effective prefetching strategy must prefetch the useful contents in a timely manner while introducing little overhead. Different prefetching strategies are proposed up to date, but no one provides optimal performance. In this paper, we examine different prefetching strategies and discuss the design tradeoffs involved when implementing these strategies. To the best of our knowledge this is the first work, which provides a performance comparison between different prefetching strategies in P2P VoD systems. On the basis of this analysis we proposed a new prefetching strategy to overcome the discrepancies in existing prefetching strategies.

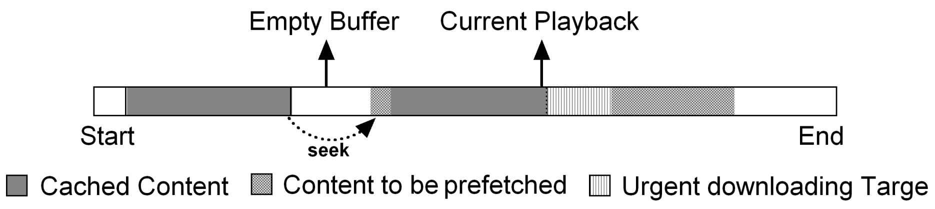


**Fig. 1. Prefetching, caching and urgent downloading**

The rest of the paper is organized in different sections. A brief related work and motivation is presented in section 2. In section 3, we provide a thorough analysis about the characteristics of user viewing behavior for different existing prefetching strategies. Section 4 discusses the proposed cooperative prefetching strategy. Section 5 illustrates the performance evaluation and section 6 presents a brief conclusion while highlighting some of the future perspectives.

## 2 Motivation and Related Works

In the past years, several researches have been proposed for multimedia caching and prefetching. The peer in a VoD system can prefetch the contents in many different ways. We first consider the simplest scheme, which is called no-prefetching [6]. Under this technique, each peer obtained the contents at streaming rate and don’t prefetch contents for seek operations. The seek operation by a user results in increased latency due to runtime prefetching. This technique increases server stress because neighboring peers don’t have the desired contents and most requests are satisfied by the server. The random prefetching [3] is used to prefetch the data in local cache before a seek operation is carried out. Rather than waiting for a cache miss to perform a prefetch, random prefetching anticipates such misses and issues a fetch to local cache in advance. The scheduler is responsible for randomly prefetching segments in periodic intervals. Although caching of data is considered in random prefetching but prefetching of data in unpredictable user behaviors has not been addressed. As a result more useless segments occupy the local cache.

The popularity aware prefetching technique [4] exploit user access patterns for prefetching the data segments. In this technique logs are maintained by a management server (also called tracker) regarding users access pattern. The statistics gathered on user requests are used to determine the optimal number and placement of replicates for each individual video file. These popular segments are distributed among the peers participating in VoD sessions. As a result popular data segments are obtained before playback. The popularity aware prefetching techniques improves hit ratio by considering user's access patterns, however large computation are required to be performed by management server for extracting the list of popular contents. The periodic exchange of seek operation's information with management server results in additional overhead. State maintenance and data mining [5] are used as another technique for prefetching more desirable segments of video. The playback history is exchanged among a set of peers (neighboring peers) which share the closest playhead positions. This playback history provides peers a data set for performing data mining operations. Unlike popularity aware prefetching, each peer performs data mining operations locally instead of central management server. Therefore, association rule mining is used to find maximum occurring segment with respect to current position.

## 3 Prefetching Technique Analysis

We defined two types of hit ratios regarding VCR operations. $HR_r$ is the "relative hit ratio", defined as "the number of prefetching request satisfied locally". This means that the new segment pointed as a result of seek operation was already available. $HR_g$ is the "global hit ratio", defined as the number of prefetching requests satisfied by requesting segments from neighboring peers. This means that the segment pointed as a result of seek operation was not available in local buffer, and therefore a prefetch request has been made to obtain the segment from other peers in same session.

Let us suppose, each peer has a memory to store *s* segments. Note that this is the maximum number of segments that can be prefetched before the occurrence of VCR operations. Let *S* be the maximum number of segments that can be prefetched by neighboring peers. We will denote *L* = {set of segments that can be requested using VCR control}. Obviously this set contains either those segments which are not played or those segments which are played but removed from memory later on. We denote $L_i$ as the set of segment prefetched by peer *i*. Let $P_i$ be the probability that the requested segment (as a result of VCR control) exists in $L_i$. In case of data-mining based prefetching mechanism for peer *i*, we have:

$$HR_r = \frac{P_i \times s}{L_i}, \quad HR_r + HR_g = \frac{P_i \times S}{L_i} \qquad \textbf{(1)}$$

For no prefetching, there is no possibility that the requested segments exists in the set of available prefetched segments. Thus for no prefetching:

$$HR_r = 0, \; HR_r + HR_g = 0 \qquad \textbf{(2)}$$

For popularity aware based prefetching we have:

$$HR_r = \frac{s}{L}, \quad HR_r + HR_g = \frac{S}{L} \tag{3}$$

Similarly for random prefetching:

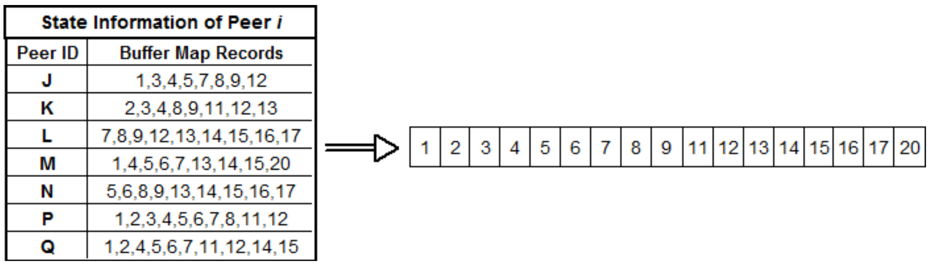
$$HR_r = \frac{P_i \times s}{L}, \; HR_r + HR_g = \frac{P_i \times S}{L} \tag{4}$$

The above equations show that both data-mining based prefetching and historical prefetching had better hit ratio comparatively. However, historical prefetching is based on a larger data set which didn't represent the user's behavior in a particular session. On the other hand, data mining based techniques prefetched the popular contents consumed by peers having closest playhead distance.

## 4 Cooperative Prefetching Technique

We observed a tradeoff between user behavior and overhead in different aforementioned prefetching techniques. Moreover either the management server (in case of popularity aware) or each peer (data mining based techniques) has to perform necessary computation which increase computational overhead. To overcome the above mentioned problem, we proposed "*cooperative prefetching technique*". We used a tree based overlay structure similar to P2Cast [7] in which peers are organized into different session according to their playhead position.

In cooperative prefetching technique, peers sharing the same session will exchange the *state information* with each other in periodic interval. This state information is the buffer map of available segments and the current play head position. For the management of buffer, each peer caches the latest 3 minutes of the video played. Apart from this each peer also holds the initial 3 minutes of video and never replaces this part during its existence in the network. This is due the "impatient" behavior of audience which scans through the beginning of videos to quickly determine their interest. The results indicates that caching the first few minutes of video is sufficient to serve 50% of all users session [1]. The format of the buffer map is *[PeerID, Playback segments, Current Playhead, Time stamp],* where PeerID is the peer's IP address, Playback segments refers to the record of segments the peer plays after it generates the last state-messages. Instead of exchanging the complete record of available segment, each peer only sends *playback segments* in order to avoid extra overhead. *Time stamp* is the time when peer sends the state information. On receiving a state message, a peer performs relevant operations before it forwards the message to its neighbors. If peer 1 receives a state information message from peer 2, it compares the time stamp of current message with earlier time stamp. If the current time stamp is greater, then the state information record is updated. Once the state informations are collected from all peers (in same session) each peer creates a table of available segments in that particular session. Fig. 2 shows the state information table received by a particular peer *i*. Each peer performs the necessary computation to remove redundancy and creates a list of available non redundant segments in the session.



| State Information of Peer *i* | |
|---|---|
| **Peer ID** | **Buffer Map Records** |
| **J** | 1,3,4,5,7,8,9,12 |
| **K** | 2,3,4,8,9,11,12,13 |
| **L** | 7,8,9,12,13,14,15,16,17 |
| **M** | 1,4,5,6,7,13,14,15,20 |
| **N** | 5,6,8,9,13,14,15,16,17 |
| **P** | 1,2,3,4,5,6,7,8,11,12 |
| **Q** | 1,2,4,5,6,7,11,12,14,15 |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 20 |

**Fig. 2.** Removing Segment Redundancy

In case of Fig. 2 missing segments like 10, 18, and 19 would be requested from far neighbors (peers in other session) depending on current playhead position. This request is made to either far neighbors or server (if there is no response from other peers). As a result, those rare segments are obtained from other session that didn't exist in the current session. Later on, if a seek operation is carried out and the segment is available in the same session, it will take less time to acquire it from neighbor peers instead of server or far peers. It is important to note that each peer also prefetch the segments near to its playhead position as an *urgent downloading target.* In our case each peer prefetch the next 20 seconds of video segments as urgent downloading target. Apart from these segments, remaining segments are prefetched using cooperative prefetching strategy.

## 5 Performance Evaluation

We organized the peers in an overlay structure proposed in P2Cast. In P2Cast, peers are distributed into sessions according to their playhead position. We used the BRITE universal topology generator in the top-down hierarchical mode to map the physical network. Each topology consists of 3 autonomous systems each of which has 10 routers. All AS are assumed to be in the Transit-Stub manner. The delay on the access links is randomly selected between 5 to 10 ms. The incoming and outgoing bandwidth of peers varies between 512 kbps to 5 Mbps and is uniformly distributed throughout the network. We deployed a single media source and the uplink bandwidth of media source is 5Mbps. When ever a peer receives the contents from a parent peer in the tree, it keeps tracks of the sequence number of the packets. The peer in same session exchanges the sequence number as part of state information.

### 5.1. Simulation Results

Fig. 3 and 4 shows the comparison of hit ratio and utilization ratio respectively. The obtained results suggest that cooperative prefetching increases the hit ratio and utilization ratio of the session. This is due to the reason that cooperative prefetching maximizes the availability of rare segments in the session. Both data mining and popularity aware prefetching techniques focused on certain number of segments according to user behavior, while ignoring segments rarity. That's why cooperative prefetching strategy has better performance.

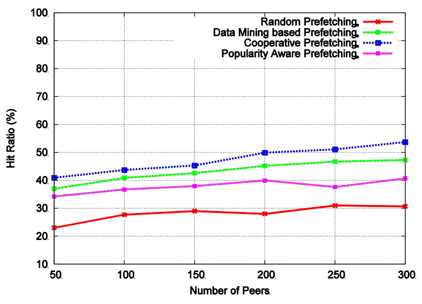


**Fig. 3.** Comparison of Hit Ratio

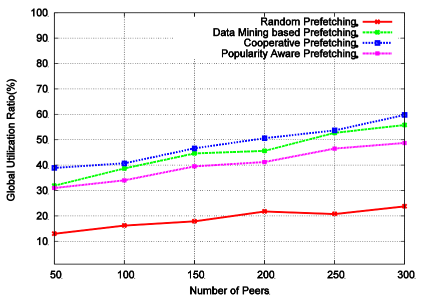


**Fig. 4.** Comparison of Utilization Ratio

## 6 Conclusion and Future Perspective

In this paper we analyzed different existing caching strategies for peer assisted VoD system. In order to provide a VCR-oriented VoD service for P2P networks, we proposed a cooperative prefetching strategy. Our strategy focuses on improving the availability of rarest contents in a session. Our proposed strategy improves the hit ratio and decrease the overhead significantly. For the future perspective, we aim to perform real test-bed evaluation for the more personalized VoD and IPTV services delivery over P2P network.

## References

[1] H. Yu, D. Zheng, B. Zhao, W. Zheng, "Understanding User Behavior in Large-Scale Video-on-Demand Systems". In Proc of EuroSys 2006.

[2] M.Mushtaq, T.Ahmed, D. Meddour, "Adaptive packet video streaming over P2P networks". In Proc of the 1st International Conference on Scalable information systems

[3] B. Cheng, H. Jin, X. Liao, "Supporting VCR functions in P2P VoD services using ring-assisted overlays". In Proc of ICC 2007.

[4] C. Zheng, G. Shen, S. Li, "Distributed Prefetching Scheme for Random Seek Support in P2P Streaming Applications". In Proc of ACM P2P multimedia streaming 2005.

[5] Y. He, Y. Liu, "VOVO: VCR-Oriented Video-on-Demand in Large-Scale P2P Networks". In Proc of IEEE Trans. Parallel and Distributed Systems vol PP, Issue 99, June.2008.

[6] C. Huang, J. Li, K.W. Ross, "Can Internet Video-on-Demand be Profitable". In Proc of ACM SigComm 2007.

[7] Y. Guo, K. Suh, J. Kurose, and D. Towsley, "P2Cast: Peer-to-peer Patching Scheme for VoD Service". In Proc of International conference on World Wide Web 2003.