First-Aid Mechanism for Peer-to-Peer Streaming

Tieying Zhang[1], Fuxin Ren[2], Xueqi Cheng[3]
1, 2 PhD Students, 3 Professor
{zhangtiey, renfuxin}@software.ict.ac.cn, cxq@ict.ac.cn
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

Abstract

A fundamental challenge in P2P streaming system is how to provide high quality of services (QoS), such as short startup time, low seeking latency and high playback continuity. This paper presents First-Aid mechanism, a reciprocal data-request scheme in which first-aid users have high priority to get data while they must provide service to normal users suspended by them. Experiments show that our scheme performs better than traditional methods. We have also implemented a scalable P2P system called Cool-Fish[1] which *supports live-streaming, VoD and file sharing services. The running results also prove the validity of our design.*

1. Introduction

In recent years, P2P streaming services have become very popular over the Internet. However, long startup, slow seeking, frequent jitters degrade the user experience. The basic reasons for these problems are not only the unpredictability of peer dynamics but also the heterogeneous service ability of peers. These issues result in data latency to the users. In order to solve these problems, many techniques are proposed. Multi-stream and patching methods are designed to support P2P streaming, but the strict relationship between peers may also degrade user experience. To address this problem, some hybrid methods are used, such as pre-fetching and network coding. A scheme in [2] is presented to support VCR by mining associations inside a video. But all the proposed approaches do not consider the data emergency status according to the playing time.

We propose a novel data-request mechanism considering the data emergency. First we divide the video into chunks, then the chunks are classified into three parts termed urgent, normal and future according to the playing time. Any peer who has urgent chunks to request is called first-aid user. And competitors are the ones who request chunks from the same parent. First-aid users have higher priority to get service from his parents than normal competitors, but then he must build partnership with the competitors and serve them. This is different from the random relationship between peers in traditional P2P system where the competitors are not always partners. However peers can obtain service from each other only when they are partners. In our mechanism, first-aid user must build partnership with his competitors as long as he makes the competitors suspended.

2. Our approach

First-Aid based data-request mechanism has two components. One is how to gain first-aid, the other is how to serve competitors to help them to avoid data latency. At the same time we adopt pre-fetching scheme to obtain chunks in pre-fetch window.

2.1 Gaining first-aid

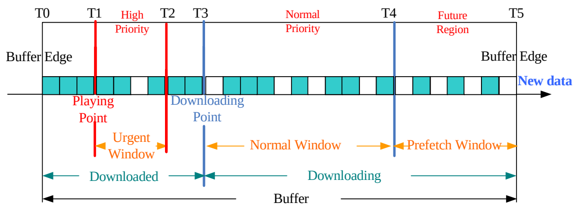

Figure1 Download buffer of a peer

As illustrated in Fig.1, download buffer consists of several parts. An important time is playing point (T_1). It is more urgent when data gets closer to playing point. According to this, buffer after playing point is divided into three sliding windows with different priorities, urgent, normal and pre-fetch. In general, user request chunks from normal window [T_3, T_4], because T_3 is the downloading point, and with chunks arriving, normal window moves on. However, when urgent window has some empty chunks which are not gained, user will request these chunks with high priority and parents serves them with high priority accordingly. This scheme can accelerate obtaining chunks when users are urgent, particularly in startup and seeking period.

2.2 Serving competitor with mutual benefit

When receiving urgent requests, parents must suspend normal requests, thus the users who send these normal requests have to wait longer. Obviously, all these requesters are competitors. To avoid such snub to normal requester, first-aid user must serve his suspended competitors when downloading his urgent chunks and if he does not have ability to serve his competitors, he will ask his neighbors to do this job. So this mechanism is helpful for the suspended users to get data as soon as possible.

3. Performance evaluation

3.1 Simulation

We have evaluated two versions of First-Aid mechanism using NS2 simulator, the first one is First-Aid(no-sc) which does not have serving-competitor (sc) mechanism, the second is First-Aid(sc) with sc mechanism. In order to compare our scheme with traditional approaches, we implement BiToS [3] which uses popular Bittorrent algorithms and adds sliding window to support live-streaming. We modify it to support VoD service. In our simulation, we set the video time to 3600s, playback rate to 1Mbps. We generate a transit-stub topology including 260 routers using GT-ITM, and we develop a tool to separate transit nodes from stub nodes. The delay between any two routers is 10ms-50ms. We generate 2000 users attached to stub routers following uniform distribution. The user arrival follows a Poisson distribution* as in [4], i.e.,

$$p(x;\lambda t)=\frac{(\lambda t)^{x}}{x!}e^{-\lambda t},x=0,1,2,...,t\geq 0,$$

with average arrival time $\tau = 1/\lambda$, here we set τ =10. The on-line time of each user follows an exponential distribution with mean T = 3600 and any user has ability to serve two streams. The startup and seeking period are both buffering 10s media data. Fig.2 and Fig.3 show the startup and seeking latency. We can learn that both First-Aid schemes perform better than BiToS, particularly in seeking period, because peers can help each other with serving-competitor mechanism and obtain more chunks with pre-fetching scheme. Fig.4 plots the continuity index which is the number of chunks that arrive before or on playback deadlines over the total number of chunks. First-Aid(no-sc) slightly outperforms BiToS while First-Aid(sc) can achieve higher continuity, because suspended users can obtain chunks in time in First-Aid(sc) with serving-competitor mechanism.

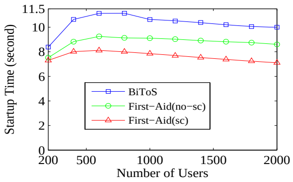

Figure2 Startup Time

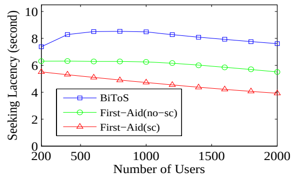

Figure3 Seeking Latency

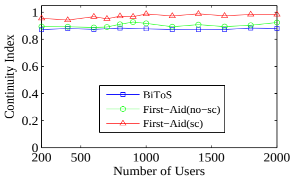

Figure4 Continuity Index

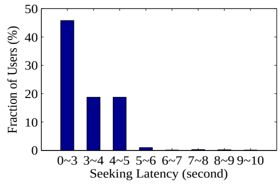

Figure5 Seeking Latency in Cool-Fish

3.2 Implementation

We have implemented a scalable P2P system (Cool-Fish[1]) to support live-streaming, VoD and file sharing services. It has been deployed in China Science & Technology Network (CSTNet) which is one of the four major networks in China. The system is able to sustain users of a par scale at a bit rate 500Kbps by only a single common server. Fig.5 shows the distribution of seeking latency in Cool-Fish. In general, the seeking latency is just around 3 seconds. More information can be found on Cool-Fish website [1]. We are going to observe more user experience for further research.

4. References:

[1] Cool-Fish website: www.cool-fish.org. 2008.

[2] Y. H. A. Y. Liu, "Supporting VCR in Peer-to-Peer Video-On-Demand," in *Proceedings of Poster session of IEEE ICNP'07*, 2007.

[3] M. Vlavianos, A. and Iliofotou, M.and Faloutsos, "BiToS: Enhancing BitTorrent for Supporting Streaming Applications," in *Proceedings of IEEE GI'06*, 2006.

[4] W. -. Yiu, J. Xing, and S. -. Chan, "VMesh: Distributed Segment Storage for Peer-to-Peer Interactive Video Streaming," *IEEE Journal on Selected Areas in Communications*, vol. 25, pp. 1717-1731, 2007.

* We have added a Poisson distribution module into NS2 to support our simulation.