Adaptive Content Distribution Network for Live and On-Demand Streaming

Yuta Miyauchi, Noriko Matsumoto, Norihiko Yoshida

Graduate School of Science and Engineering
Saitama University
Saitama 338-8570, Japan
{yuta, noriko, yoshida}@ss.ics.saitama-u.ac.jp

Yuko Kamiya, Toshihiko Shimokawa

Graduate School of Information Science
Kyushu Sangyo University
Fukuoka 813-8503, Japan
{kamiya, toshi}@nw.is.kyusan-u.ac.jp

Abstract: We have proposed an adaptive content distribution network (CDN), FCAN (Flash Crowds Alleviation Network), which changes its structure dynamically against *a flash crowd*, that is a rapid increase in server load caused by a sudden access concentration. FCAN in our preceding studies responds only to static content delivery. In this paper, we extend FCAN to alleviate flash crowds in video streaming. Through some experiments, we confirmed that FCAN for video streaming is effective to alleviate flash crowds.

1 Introduction

When a Web site catches the attention of a large number of people, it gets an unexpected and overwhelming surge in traffic, usually causing network saturation and server malfunction, and consequently making the site temporarily unreachable. This is the “flash crowd” phenomenon on the Internet. An example of a flash crowd is Figure 1, which shows the traffic volume of Web site during the solar eclipse based on a real access log provided from “LIVE! ECLIPSE 2006” [LE06]. During the solar eclipse, the accesses from clients increased several times higher than the normal condition by flash crowds.

We have proposed an adaptive content distribution network (CDN), FCAN (Flash Crowds Alleviation Network), which changes its network structure adaptively depending on a load fluctuation against flash crowds [Pan06, Yos08]. FCAN only focused on static content delivery in our preceding studies, thus in this study, we extend FCAN to video streaming.

A large amount of clients receive a sequential stream from a streaming server, therefore network traffic is concentrated on a specific site on the Internet. To assure resilience in

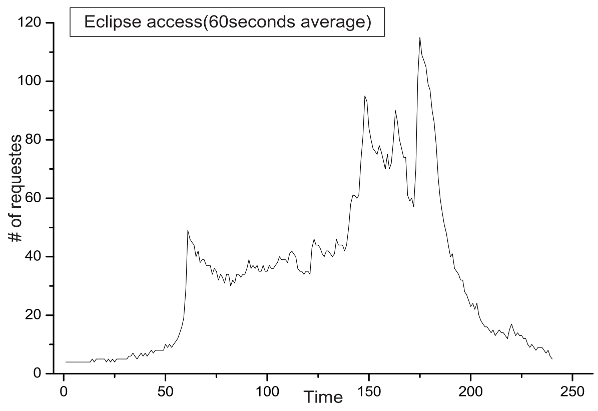

Figure 1: Flash Crowds on “LIVE! ECLIPSE 2006” site

P2P video streaming, the FCAN framework is thought to be promising. There is a survey paper on resilience in P2P video streaming [Abb11], however, there is no system which changes its network structure dynamically according to load fluctuation.

In this paper, we describe FCAN's extension. It uses Apple's HTTP Live Streaming [AD11a] for stream segmentation and distributed delivery. This paper is organized as follows: Section 2 provides a brief overview of HTTP Live Streaming. Section 3 presents an overview of our previous FCAN for static content delivery. Section 4 gives FCAN's extending design for video streaming. Section 5 describes preliminary experiments with a simple prototype. Section 6 contains some concluding remarks.

2 HTTP Live Streaming

Conventional standard streaming protocols such as progressive download and real-time streaming do not allow switching of stream source servers on the client side dynamically. Therefore, massive accesses from clients concentrates on a particular site on the Internet. Accordingly, the server and its surrounding network choke up, and a flash crowd occurs.

In order to resolve this problem of conventional protocols, Apple has introduced a new protocol for video streaming, HTTP Live Streaming (also known as “HLS”). It has been proposed as a standard draft for the Internet Engineering Task Force [Pan11]. Figure 2 shows an overview of this protocol.

The server starts providing a video stream with the following procedure: (1) Encodes an audio/video inputs; (2) Divides the encoded stream into a set of media segments (“.ts” files), and makes an index (“.M3U8” file) which refers them; (3) Delivers them to clients using HTTP on the Internet.

This protocol enable a client to switch the source server dynamically as opposed to the

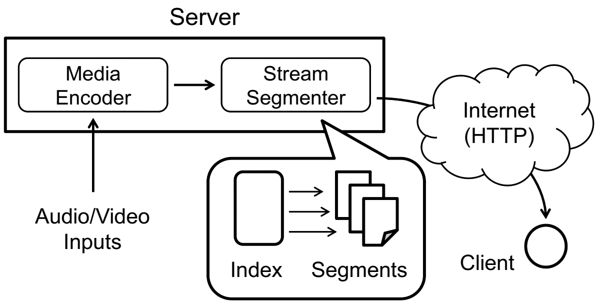

Figure 2: HTTP Live Streaming Overview

conventional streaming protocols. The delivery system archives load distribution easily with additional servers.

As another key feature, HTTP Live Streaming supports "adaptive bitrate." The server provides alternative streams with different quality levels of bandwidths, so as to enable a client to optimize the video quality according to the network situation, as the load on the network and CPU, both on the server side and the client side, fluctuates on a frequent basis.

3 FCAN

FCAN is an adaptive CDN which takes the form of C/S or CDN depending on the amount of accesses from clients. Specifically, in the C/S mode, a server provides contents to clients as in a traditional C/S. In the CDN mode, when the server detects a coming of a flash crowd, volunteer cache proxies in the Internet construct a temporary P2P network and provide the content on behalf of the server. These volunteer proxies are recruited in advance out of providers and organizations. In case servers in such providers and organizations suffer from flash crowds, they will be helped by other volunteer proxies. FCAN is built upon this mutually-aiding policy. Figure 3 shows an overview of FCAN.

In our preceding studies, we summarized some researches to alleviate flash crowds [Yos08]. These researches are divided into three categories: server-layer, inter-mediatelayer and client-layer solutions, according to typical architectures of networks. FCAN is an intermediate-layer solution, which employs an Internet infrastructure of cache proxies to organize a temporal P2P-based proxy cloud for load balancing. However, FCAN has some extensions with some dynamic and adaptive features. Our FCAN studies achieved very promising results regarding static content delivery on the real Internet [Miy11].

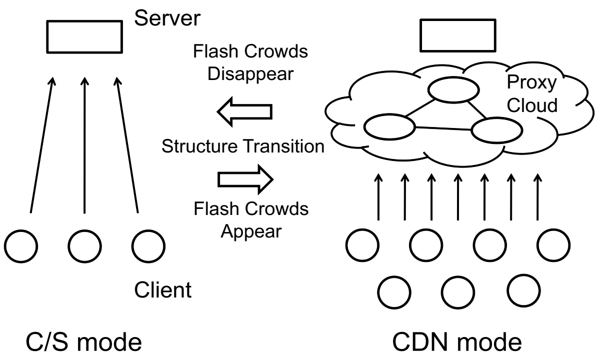

Figure 3: FCAN Overview

4 FCAN for Streaming

4.1 Structure Transition

First, We changed the behavior of structure transition in the previous FCAN design.

The server and the cache proxies in the proxy network always monitor the amount of accesses they receive from clients and evaluate the load on the network. The system switches to the CDN mode if all nodes' loads are higher than a certain threshold, and switches back to the C/S mode if lower. Each cache proxy sends its own load information to the server periodically, and the server determines whether to perform structure transition.

We use two thresholds to prevent "thrashing" between the two mode. The threshold for transition from the C/S mode to the CDN mode is set to higher than the one for transition from the CDN to C/S.

In peaceful times, the conventional C/S architecture satisfies most of the client requests. A server and cache proxies, both of which comprise FCAN, do little more than what normal ones do. When a flash crowd comes, the server detects the increase in traffic load. It triggers a subset of the proxies to form an overlay, through which all requests are conducted. All subsequent client requests are routed to this overlay.

The server-side procedure is outlined as follows: (1) Selects a subset of proxies to form a CDN-like overlay of surrogates, and builds a distribution tree; (2) Pushes the index file and stream segments to the node of the distribution tree, so as to meet the real-time constraint of the video streaming; (3) Prepares to collect and evaluate statistics for the object from the involved proxies, so as to determine dynamic reorganization and release of the overlay.

The proxy-side procedure is outlined as follows: (1) Changes its mode from a proxy to a surrogate (or, in the strict sense, a mixed mode of a forward proxy and a surrogate); (2) Stores flash-crowd objects (except the index file) permanently, which should not expire until the flash crowd is over; (3) Begins monitoring the statistics of request rate and load,

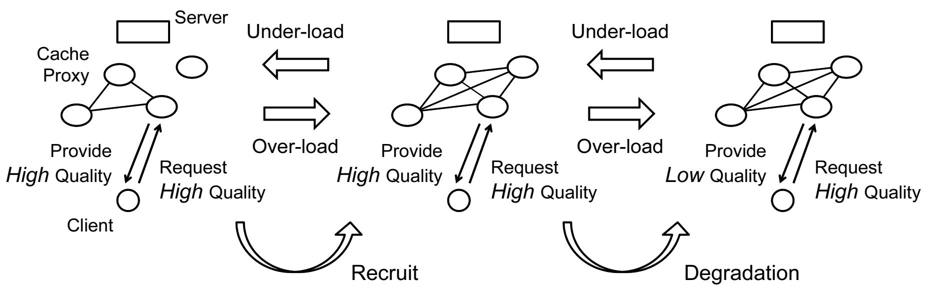

Figure 4: Handling of load increase in CDN mode

and reporting them to the server periodically.

In the live streaming, the index file is updated periodically, therefore the server monitors its composition, and pushes the segments at an appropriate time. Meanwhile, when the proxy is released by the server, it discards the index so as to keep the consistency among the nodes.

When the member server detects the leaving of the flash crowd, the involved proxies are dismissed one by one with the following procedure: (1) The server notifies the proxy to be dismissed; (2) The server requests the related proxies to modify the relation of connection; (3) The proxy changes its mode from a surrogate to a proxy.

The CDN-like overlay transits back to the normal C/S mode when all the proxies are dismissed. They are not all dismissed at once, since the low load may be just temporary, and the system should therefore remain in the anti-flash-crowd mode for a while.

4.2 Dynamic Resizing and Quality Restriction

The proxy network is a pure P2P network. Therefore, it is highly fault-tolerant and scalable. Unlike traditional P2P systems, it does not include clients into the network itself in order to assure reliability and security.

FCAN resizes a scale of the proxy network depending on a load fluctuation adaptively in order to avoid troubles such as server down by massive access concentration. Figure 4 shows how the system works in the CDN mode.

When the server detects a coming of flash crowds, it forms a temporary proxy network as shown in the lefthand side of Figure 4. If the initial network cannot handle increasing an amount of accesses, the server recruits a new member proxy one by one as shown in the middle of Figure 4.

If the server cannot recruit temporary proxies any more, it degrades the quality of the video stream as shown in the righthand side of Figure 4. For example, in the situation that the system provides video streams of two different qualities, high and low, it delivers the low quality content as substitute for the high quality one under this quality restriction. The network occupancy per client decreases so that the server can alleviate the load of whole delivery network.

If the proxy network can easily handle all the incoming loads, the server may lift the restriction of the stream quality at first. After the derestriction, it releases temporarily-recruited proxies one by one until all proxies are dismissed. Finally, the system all turns back to the normal condition.

4.3 Access Redirection

In our preceding studies, FCAN uses DNS-based redirection, i.e. the authoritative DNS server redirects an access to an appropriate node depending on the network structure. We use TENBIN [Shi00] for the authoritative DNS server. It is a high-performance DNS which allows server selection policies and DNS lookup entries to be changed dynamically. DNS-based redirection works transparently to users, however, we confirmed "cache effect" problem which is caused by some DNS servers somewhere in the world which make caches of the address resolution at their own discretion.

In this study, we make the client access redirect to an appropriate server through the *mediator*. The mediator works on the same machine as the client software and relays requests from the client to the servers. It handles client requests with the following procedure: (1) Receives a list of working servers from the origin server; (2) Receives the content from a certain server in the list, and provides it to the client; (3) When the mediator get a request from the client next time, it receives new list from a certain server; (4) Return to (2).

Using the mediator, we eliminate the cache effect problem, and even utilize geographical information-based redirection for example. While the mediator works non-transparently to users, we expect that the function of the mediator can be implemented in browser cookies in the future.

5 Preliminary Experiments

We conducted some preliminary experiments on a real network with a prototype of the system. Figure 5 shows an overview of the experiments.

In our experiments, we use some hosts in Saitama University and Kyushu Sangyo University for a server, proxies, a pseudo client, and a client node. We use Apple's stream segmenter (mediastreamsegmenter) for the segmenter, and QuickTime Player for QuickTime X in the client.

The pseudo client is to trigger the FCAN's functions against flash crowds. It submits requests for randomly chosen segments to the server following the pattern shown in the Figure 6. In the rest of this section, CP1, CP2 and CP3 are the proxies shown in the Figure

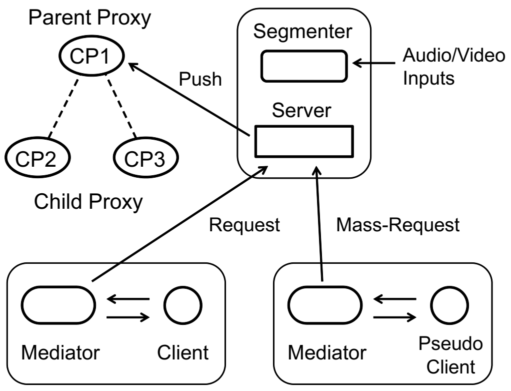

Figure 5: Experiment Environment

Table 1: Segments for On-Demand Experiment

Quality	Resolution	Average size	Duration
High	480×360	960 [KB]	10 [sec]
Low	320×240	410 [KB]	10 [sec]

5.

We made two experiments with two delivery methods, live and on-demand. In the on-demand streaming experiment, the server provides high and low quality contents with adaptive bitrates. We use segments shown in Table 1, which are samples of HTTP Live Streaming in the Apple Developer's site [AD11b]. The client software, the client has a master index file indicating these two quality contents, and the QuickTime Player requests segments of an adequate quality depending on load fluctuation following the master index. On the other hand, in the live streaming experiment, the server provides contents in a single quality in real time.

The server computes a load value regarding the size of requested segments. In the experiments, thresholds for load detection are defined beforehand based on some experiences. Workloads on the real Internet varies, and automatic and dynamic configuration of the thresholds is difficult. We suppose they may be configured based on the server capacity and the network bandwidth around the server.

Table 2 shows the time-line of the live experiment. We confirmed that the structure transition and dynamic resizing were performed depending on load fluctuation.

Table 3 shows the time-line of the on-demand experiment. In addition to the result of the live streaming, the stream quality was limited during server load growth.

Table 4 shows the sequence of the stream segments which the client played in the on-demand experiment. The client consistently requested the high quality contents until the

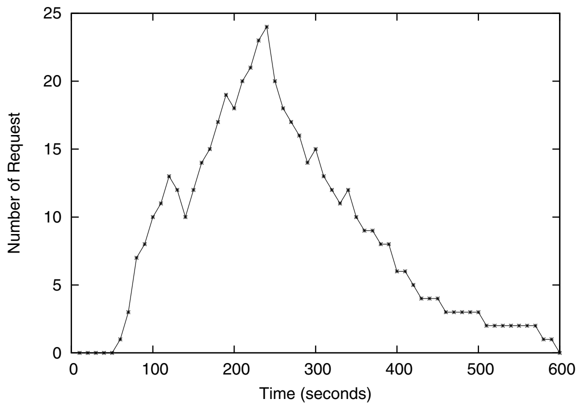

Figure 6: Request Pattern of Pseudo Client

Table 2: Time-line of Live Experiment

Time	Actions
0 [sec]	Start experiment Pseudo client request start
30 [sec]	Client request start
130 [sec]	Structure transition to CDN Recruit the core proxies (CP1, CP2)
220 [sec]	Recruit the additional proxy (CP3)
490 [sec]	Dismiss the additional proxy (CP3)
550 [sec]	Structure transition to C/S Dismiss the core proxies (CP1, CP2)
600 [sec]	End experiment

end of the experiment. In the on-demand streaming, the segmentation is done before the beginning of the experiment and the composition of the index does not change, therefore, clients received some number of segments before playback. After the number 27, the server degraded the stream quality for load alleviation, so the client received low quality segments as substitutes for the high quality ones. As the server load decreased, the server lifted the restrictions on the quality. After the number 45, the client herewith received high quality segments as required again. During the experiment, no malfunctioning, such as interrupt in the playback, was observed when the quality changed.

Figure 7 shows the load transitions of the server with FCAN and without FCAN in the on-demand experiment. The case of the server with FCAN shows the average loads of member nodes in the distribution network. The first peak at the 40th second shows that "buffering" was done when the client started the playback as mentioned above. The load on the server exceeded the higher threshold at the 130th second, then structure transition

Table 3: Time-line of On-Demand Experiment

Time	Actions
0 [sec]	Start experiment Pseudo client request start
30 [sec]	Client request start
130 [sec]	Structure transition to CDN Recruit the core proxies (CP1, CP2)
220 [sec]	Recruit the additional proxy (CP3)
250 [sec]	Quality degradation
430 [sec]	Quality improvement
490 [sec]	Dismiss the additional proxy (CP3)
550 [sec]	Structure transition to C/S Dismiss the core proxies (CP1, CP2)
600 [sec]	End experiment

Table 4: Playback Time-line of Client

Segment Name	Size	Quality
fileSequence0.ts	926 [KB]	High
fileSequence1.ts	946 [KB]	High
fileSequence2.ts	950 [KB]	High
...	...	...
fileSequence26.ts	958 [KB]	High
fileSequence27.ts	414 [KB]	Low
fileSequence28.ts	410 [KB]	Low
...	...	...
fileSequence44.ts	414 [KB]	Low
fileSequence45.ts	958 [KB]	High
fileSequence46.ts	958 [KB]	High
...	...	...
fileSequence60.ts	967 [KB]	High
fileSequence61.ts	958 [KB]	High
fileSequence62.ts	963 [KB]	High

to the CDN mode occurred so as to alleviate load concentration. However, the load on the member nodes continued to increase even after the transition, a new member proxy was recruited at the 220th second, and additionally the quality of segments was degraded at the 250th second, and consequently the server withstood the heavy load condition.

On the contrary, in the case of the server without FCAN, we observed that load values were far exceeding ones of the server with FCAN consistently. We, therefore, confirmed that FCAN's features against flash crowds were performed as a result of the increase of client

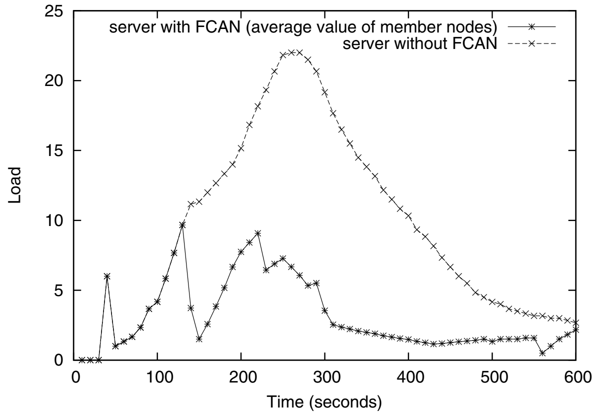

Figure 7: Comparison of load transitions in On-Demand Experiment

requests, and FCAN archived dynamic load balancing. We obtained equivalent results also in the live streaming experiment.

6 Conclusion

In our preceding studies, FCAN only focused on the static content delivery, however, flash crowds occur also in the video streaming. In order to alleviate flash crowds in the video streaming, both live and on-demand, FCAN adopts a new features such as *dynamic resizing* and *quality restriction* so as to raise a resilience of the system. In this paper, we proposed FCAN's extension for video streaming and demonstrated a prototype of the system on the real Internet. Through some experiments, we confirmed that FCAN's extension works effectively to alleviate flash crowds.

We are still at a starting point toward practical implementation and promotion of FCAN. Future research directions include: (1) quality guarantee in the CDN mode in the situation that multiple proxies deliver the same stream content, (2) appropriate thresholds assignments, and (3) implementation of dynamic access redirection using browser cookies.

References

[Abb11] O. Abboud, et. al. Enabling Resilient P2P Video Streaming: Survey and Analysis. *Multimedia Systems*, Vol:17, Springer, pp.177–197, 2011.

[AD11a] Apple Developer. HTTP Live Streaming. http://developer.apple.com/resources/http-streaming/, 2011.

[AD11b] Apple Developer. Bip Bop All. http://devimages.apple.com/iphone/samples/bipbopall.html, 2011.

[LE06] LIVE! ECLIPSE. http://www.live-eclipse.org/, 2006.

[Miy11] Y. Miyauchi, et. al. Preliminary Study on World-Wide Imprementation of Adaptive Content Distribution Network. *Proc. Workshop on Self-Organising, Adaptive, Context-Sensitive Distributed Systems* , 11 pages, 2011.

[Pan06] C. Pan, et. al. FCAN: Flash Crowds Alleviation Network Using Adaptive P2P Overlay of Cache Proxies. *IEICE Tr. Comm.*, E89-B(4), pp.1119–1126, 2006.

[Pan11] R. Pantos. IETF Internet Draft: HTTP Live Streaming. http://tools.ietf.org/html/draft-pantos-http-live-streaming, 2011.

[Shi00] T. Shimokawa, et al. Flexible Server Selection Using DNS. *Proc. Int. Workshop on Internet 2000 (in IEEE-CS 20th Int. Conf. on Distributed Computing Systems)*, pp.A76–A81, 2000.

[Yos08] N. Yoshida. Dynamic CDN against Flash Crowds. *Content Delivery Networks* (R. Buyya, et al., eds.), Springer, pp.277–298, 2008.