CS 237: Interactive Movie Streaming Service

Wei Pan, panw4@uci.edu
Yi Zhou, zhouy46@uci.edu
Xiaomi Liu, xiaoml6@uci.edu

Motivation

- Inspired by Netflix’s movie *Black Mirror: Bandersnatch*, we want to provide interactive movie streaming service.
- *Black Mirror: Bandersnatch* is an interactive film in the science fiction anthology series *Black Mirror*, produced by Netflix in 2018. The audience can make decisions for the main characters and change the story. That is, they can choose the video clips of the movie to watch.

Goal

The objective of this project is to implement a video streaming service that:

- provide a basic video streaming function,
- dynamically redistribute the video clips in caches based on the user requests for fast retrieval, and
- provide an interactive user-interface that allows the users to make selections of the video.

Related Works

Distributed video storage
Berkeley Distributed VOD System uses Compound Media Object (CMO), which allows a movie to be composed of many subjects, which may in turn be shared with other movies and cache management technique like least recently used (LRU) or random replacement.

VMesh is a distributed segment storage for peer-to-peer interactive video streaming.

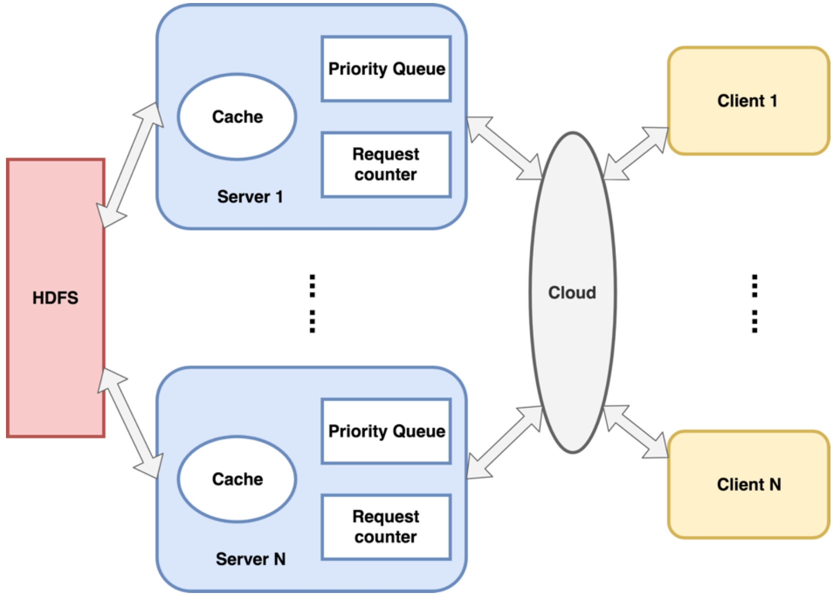
HDFS stores file system metadata and application data separately. The metadata is stored on a dedicated server, called the NameNode. Application data are stored on other servers called DataNodes.

Rate Adaptation
In order to prevent client buffer from under-flowing or over-flowing and improve the user experience of multimedia streaming services, C. Liu et al. propose a receiver-driven rate adaptation algorithm for adaptive HTTP streaming. When probing the spare network capacity, a stepwise switch-up method is used to switch to a higher representation. Upon detecting network congestion, an aggressive switch down method is deployed to prevent playback interruptions.

Implementation

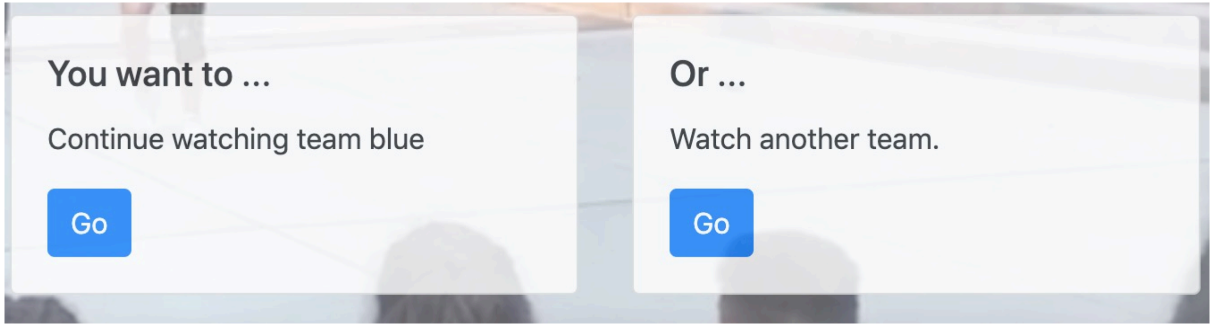
- The video data is stored on server using HDFS. We use servlet in Java to receive and respond to requests from web clients.

- We use caching to speed up the loading of most frequently requested video clips. For each video clip, we keep track of the number of times it is requested by the user in the server and keep a priority queue of video clip IDs with top 5 number of requests.



Implementation Cont’d

- The user interface is implemented in HTML 5. When a user finish watching a video clip, several choices with hints or descriptions regarding the content of the following videos are presented on the page.


You want to ...

Continue watching team blue

Go

Or ...

Watch another team.

Go

Evaluation

- We use three MacBook Pros to test our video streaming service, with one of them being the server and the other being the clients. Both clients successfully connect to the web service and can make selections for video clips independently.

- We also calculated the video clip loading time for each request from client to evaluate the performance of our dynamic caching approach.

Request No.	Requested video clip ID	Loading time (ms)
1	2	179
2	3	194
3	4	185
4	3	0
5	5	181
6	2	0

Future Work

The future extension of this project will be dynamic replication across different servers:

With the statistics kept for the frequency of each video clip, we can replicate them on multiple servers according to the frequency. The more servers that have this video clips, the broader bandwidth and less response time the client is going to get for streaming.

(Due to the limitation of equipment and environment, we have only one computer as server and another two as clients, all of which are connected in a stable WiFi network. Thus, it is not possible to measure the performance gain from the dynamic replication.)