CS330 Final: SongTube VM Design

V2.0 - Demand Page Extension/ Shared memory/ New eviction policy

2020130 김유준

## 0. Introduction

Songtube는 video streaming service로 현재 VM thrashing이 자주 일어가고 있기 때문에 이를 방지하기 위한 새로운 VM System을 design한다.

## 1. Workload관점에서 Design.

### 1-1. Demand Page의 확장

우선 한 동영상을 시청할 때, 앞에서부터 시청하는 경우가 많으며, 때문에 대부분의 스트리밍 서비스들은 앞으로 상영될 몇 초 분량의 파일만 사용사의 기기에서 다운 받도록 한다. 서버의 관점에서도 메모리가 부족한 시점에서 시청자에게 전달해줄 부분만 메모리에 올라와 있으면 된다. 그렇다면 사용자에게 전달해줄 데이터를 가지고 있는 페이지를 언제 메모리에 올려야 할까.

Demand paging 기법을 사용하게 되면, 사용자에게 데이터를 전송하려고 할 때 page fault가 일어나게 되어 데이터를 메모리에 올리게 되고 전송을 할 수 있게 된다. 이러한 구현은 파일 전체를 동시에 메모리에 올릴 필요가 없어 한 유저가 동영상을 스트리밍하는데 필요한 Page수가 줄어든다. 이미 전송을 마친 데이터는 Evict될 수 있다.

하지만, 동영상의 크기와 Page 수를 고려했을 때, 하나의 동영상을 처리하기 위해 너무 많은 Page fault가 일어남을 알 수 있다. 일반적인 상황에서 사용자가 동영상을 시청하고 있다면 시간에 따라 연속적으로 데이터를 전달해야 할 것을 예상할 수 있다. 때문에 동영상 파일 데이터에 대해 page fault가 났을 때 뒷부분에 해당하는 데이터를 가지고 있는 page도 함께 메모리로 가져오는 것을 생각할 수 있다. **일반적인 page size는 4KB인데, 한번 파일 동영상 데이터를 포함한 page에서 page fault가 날 때 연속한 4MB의 데이터를 가져온다**면(즉 1000개의 page를 가져온다면), page fault의 빈도를 줄일 수 있다.

이 demand paging 확장 기법은 **유저의 interactiveness관점에서도 의미**있다. 만약 유저가 긴 동영상을 시청하려고 했을 때, 전체 동영상을 모두 메모리에 올려놓는 원래의 기법에 따르면, 디스크에서 파일을 읽어오는데 긴 시간이 걸릴 것이다. 반면, 앞으로 필요한 일부의 page만 가져오게 된다면 사용자 입장에서의 반응속도를 개선할 수 있다. 다만 I/O가 더 자주 일어난다는 단점이 생기는데, 한번에 I/O를 통해 받아올 데이터의 양을 시뮬레이션을 통해 잘 정하는 것이 중요하겠다.

구체적으로 한번에 받아오는 양이 적다면 I/O를 자주 해야하고, disk search를 위해 사용되는 시간이 많다. 사용자가 전달받는 데이터의 스트리밍을 마쳤을 때까지 다음 데이터가 준비되어있지 않을 수 있다. 한번에 받아오는 데이터 양이 많다면, 데이터를 저장하기 위해 기존의 page를 많이 evict해야하고, 데이터를 읽는데 더 많은 시간이 걸리지만 I/O 횟수가 줄어든다.

1-2. 여러 유저간 공유된 파일 전송

Songtube는 이 동시에 시청하는 스트리밍 서비스이기 때문에 같은 동영상을 여러 사람이 동시에 시청하게 될 가능성이 있다. 예를 들어 2명의 사용자가 A, B가 동시에 동영상 V를 시청하고 있다고 하자. A라는 시청자가 동영상을 더 빨리 보기 시작했다면, A시청자가 동영상을 시청하면서 1-1구현에 의해 Page fault가 나면서 동영상이 메모리에 올라가게 된다. 그렇다면 B 시청자에게 동영상 데이터를 전달할 때, 메모리에 새로운 copy의 data를 올릴 필요 없이 이미 올라온 데이터를 전송하면 된다.

이는 같은 동영상은 시청하고 있는 사용자를 프로세스에서 **같은 process에서 multi-threading** 기법을 사용하여 공유된 데이터를 사용하게 하거나, 더 좋은 방법으로는 memory mapped파일을 사용하여 잦은 파일 입출력을 없앨 수 있다.

**Memory mapped file을 사용**한다면, kernel로의 전환 없이 메모리에 맵핑된 파일 데이터를 읽고 쓸 수 있으며, kernel로 전환될 때의 overhead를 줄일 수 있다. Memory mapped file은 1-1에서 기술한 demand page의 extension과 함께 사용될 수 있다. Memory mapped file은 여러 process에서 동시에 접근하도록 할 수 있어 Songtube의 특성에 적합하다.

1-3 Evict policy 관점

다만, 여기서도 동영상 전체를 계속 메모리에 가지고 있는 것은 현재 Songtube 메모리 오버헤드에 더욱 부담이 될 수 있다. 결국 어떤 page들은 evict를 해야하는데, 이 때 evict하는 policy를 잘 정해야 한다. 기본적으로 현재 memory mapped file을 demand paging으로 구현한다고 했을 때, 각 동영상의 동시에 시청하고 있는 시청자 수를 바탕으로 해당 동영상에 얼마나 많은 Page를 부여할지 결정할 수 있다.

어떤 동영상의 동시 시청자 수가 많다면, 동영상의 일부를 evict하더라도 금방 swap-in해야할 가능성이 크기 때문에 evict를 덜 하는 것이 좋고, 동영상의 동시 시청자수가 적다면, 반대의 일이 일어날 것이다. 따라서 해당 비디오 파일의 동시 시청자수에 따라 해당 파일에 관련된 페이지가 동시에 얼마나 많이 메모리에 올라갈지를 정하면 된다.

하지만 이러한 점은 LRU관점에서 이미 구현이 되었다고 볼 수 있겠다. 현재 SongTube 서버의 VM시스템은 LRU방식으로 작동 중인데, 동시에 시청하고 있는 사람이 많은 동영상에 해당하는 Page들은 자주 접근이 될 것이고 따라서 잘 Evict되지 않을 것이다.

문제는 document에도 서술되어 있듯, **unwatched parts에 해당하는 데이터가 LRU Algorithm에 의해 계속 Evict**된다는 점이다. Unwatched part는 데이터 전송에 아직 사용되지 않았기 때문에 LRU관점에서 최근에 사용되지 않은 page로 evict 대상이 된다. 하지만, 우리는 동영상 스트리밍의 특성 상 Unwatched parts가 앞으로 사용될 부분이라는 것을 안다. 이제 **1-1에서 구현한 demand page의 확장을 사용한다고 할 때,** Page reference부분도 신경을 써줘야 한다. page fault로 동영상 데이터를 가져올 때 연속한 page를 추가적으로 가져오는 demand page의 확장에서, **page fault가 일어난 page 뿐만 아니라 뒤에 함께 같이 메모리로 swap-in 되는 page또한 Reference된 것으로 처리**를 해 주도록 한다. 그러면 뒤따르는 page들이 LRU 알고리즘에 의해 evict 될 가능성이 적어진다.

이로써 동영상 스트리밍이라는 관점에서 LRU가 가지는 문제와 이를 해결하기 위한 해결방한이 제시됐다.

### 1-4. 인기 급상승 동영상 및 인기 저조 동영상 관리

여러 스트리밍 서비스에도 그러하듯 인기 급상승동영상은 여러 시청자에 의해 동시에 시청되며, 동영상이 크더라도 메모리에 전체 동영상 데이터를 다 저장해도 될 만큼 특정 소수의 동영상이 집중적으로 시청되곤 한다. Songtube에서는 **단위 시간(예를 들어 1시간)마다 업데이트 되는 인기동영상 10개를 선정**하여 집중적으로 노출한다(앱 UI상 인기동영상 카테고리가 있다고 가정하자). 이 10개 동영상은 쉬지 않고 여러 사람들에게 스트리밍 된다. 이 동영상들은 메인 메모리에 전체 동영상을 올려두는 것이 swap-in/out에 따른 overhead를 제거하는데 큰 도움이 될 것이다.

이를 위해서 **인기 급상승 동영상을 위한 memory공간을 따로 지정**해두고(예를 들어 5GB), 이 공간은 오로지 인기 급상승 동영상을 저장해 두는 용도로 사용한다. 만약 인기 급상승 동영상 10개의 크기의 총 합이 5GB를 넘는다면, 일반 동영상과 마찬가지로 관리될 텐데, 이 때도 page가 계속해서 reference되면서 swap-in/out이 자주 일어나지는 않을 것이다. 다만, 급상승 동영상을 일반 동영상처럼 관리를 한다면 동시에 너무 다양한 동영상이 시청 될 때에는 어쩔 수 없이 급상승 동영상 데이터의 일부도 swap-out될 수 있어 이를 막고자 5GB의 공간을 따로 확보하였다.

반면에 인기 저조 동영상은 정말 가끔 한 명의 사람이 볼까 말까하는 동영상을 일컫는다. 이 동영상의 경우, 누군가 시청을 하면서 메모리에 올라오더라도 앞으로 사용될 가능성이 굉장히 적기

때문에 최대한 빨리 메모리에서 Evict하는 것이 좋다. 이런 인기 저조 동영상은 주기적으로(예를 들어 1주일에 한 번) 조회수를 확인하여 선발하고, 동영상의 메타데이터로 가지고 있는다. 현재 메모리에 올리는 동영상이 **인기저조 동영상이라면**, 특정 **시청자에게 해당 동영상의 데이터를 전송함과 동시에 memory에서 evict**를 해주는 코드를 추가하면 된다.

## 2. Conclusion

이렇게 Songtube 서비스의 VM thrashing을 막기 위해 Paging기법, 파일 공유 관전, Eviction관점, 인기도 관점으로 VM을 어떻게 관리하면 좋을지를 살펴보았다.