Hiccup-free 디스플레이를 위한 CM 서버상의 부분복제를 이용한 슬롯 예약 알고리즘

최홍묵*0, 박병수**, 최명렬*
*한양대학교 전자전기제어계측공학과
**상명대학교 컴퓨터정보통신 공학부
{chmook*0,choimy*}@asic.hanyang.ac.kr, bpark**@smuc.ac.kr

Slot Reservation Alogorithm Using Partial Replication for Hiccup-free Display in CM Servers

Hong-Mook Choi*0, Byoung-Soo Park**, Myung-Ryul Choi*
*Dept. of EECI, Hanyang University
**,Dept. of Computer, Information and Telecommunications, Sangmyung University

요 약

본 논문에서는 연속 미디어에 있어서의 hiccup-free 디스플레이, startup latency와 많은 관련이 있는 데이터 배치 방법 및 startup latency를 줄이기 위한 복제 기술(완전복제, 선택복제, 부분복제)을 검토하고 그중 부분복제 기술에 있어서 그룹에 비어있는 슬롯이 없는 경우 request migration이 불가능하여 hiccup이 발생할 수 있는 경우에 그룹의 슬롯을 예약함으로써 hiccup을 줄일 수 있는 알고리즘을 제안하였다. 이 슬롯 예약 알고리즘은 시스템의 처리율은 다소 줄었지만 hiccup-free 디스플레이가 가능하도록 하였다.

1. 서 론

최근 정보통신 분야의 기술적인 발전으로 Continuous Media(CM;연속미디어)의 저장, 검색, 디스플레이 또한 많은 발전을 해왔다. 연속미디어 시스템은 엔터테인먼트 산업, 교육용 어플리케이션, 원격 화상회의, 전자 도서관 등 그 활용 범위가 광범위하다[1]. 이러한 시스템에서 가장 중요한 것은 다중 요청들을 처리하는 방법이다. 즉, 사용자들은 영화 등과 같은 객체를 요구하고, 적절한 지연시간 내에 볼 것을 기대한다. 지연시간은 request가 도착한 때부터 시스템이 디스크로부터 객체 읽기를 초기화하는 시간 간격으로 정의한다. 지연 요소 중에서 얼마나 많은 사용자들에게 동시에 서비스할 수 있는가를 나타내는 입출력 대역폭과 서버에 저장할 수 있는 객체의 수를 결정하는 저장 용량은 매우 중요한 자원이다. 저장 공간과 대역폭을 최대로 이용하기 위하여 객체의 블록들은 디스크 배열, 즉 데이터 배치에 신중해야 한다[2, 3].

사용자는 적절한 지연시간 뿐만 아니라 중단과 지연이 발생하지 않는 즉, hiccup이 발생하지 않는 디스플레이를 원한다. 본 논문에서는 부분복제 기술을 이용하여 그룹의 슬롯을 예약 방식으로 hiccup을 줄이는 알고리즘을 소개하였다.

본 논문의 구성은 2장에서는 Hiccup-free 디스플레이와 데이터 배치에 대해서 설명하고, 3장에서는 startup latency를 줄이기 위한 복제기술(완전복제, 선택복제, 부분복제)에 대해서 소개하고 부분복제 기술에서 그룹의 비어있는 슬롯이 없는 경우 request migration이 불가능하여 hiccup이 발생할 수 있는 경우에 그룹의 슬롯을 예약함으로써 hiccup을 줄일 수 있는 알고리즘을 제안하였으며, 4장에서는 결론 및 향후 연구 계획에 대해서 논하였다.

2. Hiccup-free 디스플레이와 데이터 배치

연속 미디어의 검색과 디스플레이는 기억장치, 스케줄링, 데이터의 전송, 다수의 사용자가 자원을 공유하는 방식 등의 실시간 제약조건에 의해 좌우된다. 만약, 실시간 제약조건을 만족하지 못한다면 디스플레이는 비디오에서의 지터(jitter)와 오디오에서의 랜덤 노이즈와 같은 결과로 중단(disruption)과 지연의 분제점이 발생한다. 이 중단과 지연을 hiccup이라 한다. 따라서 연속 미디어 서버는 hiccup-free 디스플레이를 제공하는 것이 바람직하다[4].

디스크 드라이브는 대역폭 R_D를 제공하고, 모든 객체는 같은 디스플레이율 R_C를 갖는다고 가정하자. 객체 X의 연속 디스플레이를 지원하기 위해, X를 n개의 같은 크기의 블록으로 분할(X_0, X_1, X_2, … , X_{n-1})하고, 블록은 디스크에 연속적으로 배치한다. 블록을 디스플레이 하기 위해 요구되는 시간은 시간 주기($T_p = B/R_C$)로 정의된다. 블록은 디스크 드라이브에 round-robin 방식으로 할당된다. 객체의 첫 번째 블록은 임의로 선택된 디스크(d_i)에 배치되면, j번째 블록은 $d_{(i+j-1)} \bmod d$ 식에 의해 배치된다[5]. 예를 들면, 그림 1에서, 시스템은 6개의 디스크

드라이브로 구성되어 있고, 객체 X의 블록 배치는 디스크 d_2에서 시작하고 X_1은 d_3, X_2는 d_4에 배치된다.

시간 주기에서 슬롯의 수(N)는 디스크 드라이브에 의해서 지원되는 동시 디스플레이의 최대 수로 정의한다. d개의 디스크를 가지고 있는 경우, 디스크는 시간 주기에서 동시에 d개의 디스크를 액세스하므로, 시스템은 시간 주기에서 d× N 시간슬롯을 유지한다. 다중-디스크 시스템에서 연속 디스플레이를 지원하기 위해서 request는 한 그룹에 위치하고 각각의 그룹들은 라운드-로빈 방식으로 디스크를 찾아간다(그림 1). 객체 X에 대한 요청이 도착할 때, 시스템은 X의 첫 번째 블록이 존재하는(그림 1. G_5 & d_2) 디스크 드라이브를 지금 액세스하고있는 그룹에서 이용할 수 있는 슬롯을 찾는다. 시스템에 d개의 디스크가 있고 각 디스크가 N개의 동시 디스플레이를 지원할 수 있다면 그때 시스템의 최대 처리율 m은 d× N이고, 최대 시작 지연시간은 T_p× d이다[4].

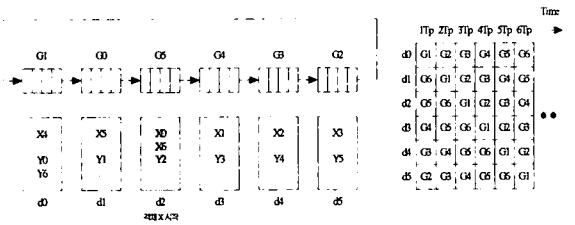

그림 1. 그룹의 순환

3. 복제 기술 및 부분 복제 기술에서의 슬롯 예약

3.1. 복제 기술

시스템의 startup latency를 줄이기 위한 기술에는 다음과 같은 기술이 있다.

완전복제(Full Replication: FR) 기술은 데이터베이스의 전체 객체를 복제하는 것으로 모든 블록은 같은 수의 복사본을 갖고 있다. 객체 X의 원본(original copy)을 X의 주 사본(primary copy) X^p라고 하자. X의 모든 다른 사본을 X의 부 사본(secondary copy)라고 나타낸다. 시스템은 객체 X에 대한 r개의 부 사본으로 구성되어 있다고 할 수 있다. 각 X의 부 사본은 $X^i(1 \le i \le r)$로 표시한다. X의 인스턴스의 수는 X의 사본의 수이다. 즉, r+1(r개의 부 사본 + 1개의 원본)이다. X의 주 사본의 첫 번째 블록이 임의의 디스크 d_i에 할당된다면(d_i는 X_0^p를 포함), 복제의 이득을 최대화하기 위해 사본 X^j의 할당은 디스크 $[(d_i + \frac{id}{r+1}) \bmod d]$에서 시작된다[4].

객체 X가 두 개의 인스턴스를 갖고 있다고 가정하면, X의 주 사본의 첫 번째 블록(X_0^p)이 포함된 디스크와는 다른 디스크에 X_0^1의 할당을 시작하며 이 경우, 최대 startup latency를 1/2로 줄일 수 있다. 또한 예상되는 startup latency도 줄일 수 있다.

선택복제(Selective Replication: SR) 기술은 실제 어플리케이션에서 각 객체마다 액세스 빈도가 다르다는 점을 감안해서 완전복제보다 저장 공간의 효율을 높인 기술이다. 즉, 저장 공간의 큰 증가 없이 startup latency를 충분히 줄일 수 있는 기술이다.

사용자가 요구하는 빈도가 다르기 때문에 이 기술에서는 객체 j의 인스턴스의 수를 결정하는 것이 가장 중요하다. 예를 들면, 소량의 최근 개봉영화는 많은 사용자들이 그 영화를 보기 원한다. 즉, 그 만큼 많은 수의 복제가 필요하다는 뜻이다. 어떤 객체에서 최적의 인스턴스 수를 결정하는 방법에는 Hamilton method, Devisior method 등이 있다[6].

부분복제 (Partial Replication : PR) 기술은 연속 미디어에서 객체의 크기와 어플리케이션의 이용 가능한 저장 공간이 제한되어 있는 것을 고려하여 제안된 것이다. 이 기술은 완전복제와 선택복제처럼 객체의 모든 블록을 복제하는 것이 아니라 각 객체의 첫 번째 블록부터 일부분만을 복제하는 기술로 완전복제와 선택복제보다 작은 저장 공간을 요구하며 startup latency를 줄일 수 있는 장점이 있다. 예를 들면, 객체 X의 10%를 10번 복제한 것은 10개의 인스턴스를 갖고 있는 즉, 저장 공간이 5배가 필요한 완전복제와 같은 시작 지연시간을 기대할 수 있다.

객체에 대한 request는 주 사본의 첫 번째 블록이 속해 있는 그룹($G_{X_0^p}$)에 빈 슬롯이 있다면 그 곳에 할당되고, $G_{X_0^p}$에 빈 슬롯이 없을 경우 $G_{X_0^1}$에 빈 슬롯이 있다면 $G_{X_0^1}$에 할당된다. 주 사본과 부 사본의 첫 번째 블록이 속해 있는 그룹에 빈 슬롯이 없을 경우 그 request는 실패가 된다. request의 할당은 완전복제, 부분복제와 유사하지만 차이점은 $G_{X_0^1}$에 할당된 request가 일부 복제된 사본의 마지막 블록에 도달하기 전에 $G_{X_0^p}$에 재배치된다. 그러나 이것은 부분 복제된 마지막 블록의 디스플레이가 될 때까지 부분 복제된 부분에 할당된 request가 주 사본에 재배치되지 않을 가능성이 있다. 그때 hiccup이 발생한다. 만약 한 디스크 드라이브가 N개의 동시 디스플레이를 지원할 수 있고 S가 객체의 평균 디스플레이 시간(서비스 시간)이라면 디스크 드라이브의 서비스 율은 N/S이다. 그러므로, 이상적으로는 객체의 첫 블록부터 S/N 부분까지 블록을 복제한다면 hiccup은 발생하지 않는다. 그러나, request가 시스템에 도착하고 떠날 때 시간의 통계적 변화율 때문에 이 기술에서 request에 hiccup이 존재 할 확률이 있다. 이 확률을 줄이기 위해, S/N보다 많은 부분을 복제해야 한다. 그 결과로 많은 저장 공간이 요구된다[4].

3.2. 부분 복제 기술에서의 슬롯 예약

임시 버퍼를 가지고 request migration은 hiccup 문제를 줄일 수 있다. 그러나 이 방법 또한 여전히 hiccup이 발생할 수 있다. 예를 들면, 그림 2에서 G_4에 있는 request는 G_1이 X의 주 사본을 액세스하는 동안 X의 부분 복제된 사본을 액세스한다. G_4가 부 사본의 마지막 블록에 도달할 때까지 G_1이 빈 슬롯을 갖고 있지 않을 때 hiccup이 발생한다. hiccup 상태를 피하기 위해 G_4에 있는 request를 G_0에 이동해서 버퍼를 사용하려고 하나 G_0역시 빈 슬롯이 없을 경우 hiccup이 발생한다.

따라서 본 논문에서는 이러한 hiccup 발생을 줄이기 위해 슬롯 예약 알고리즘을 제안하였다. 부 사본 블록이 K개(블록 $X_0^i \sim X_{K-1}^i$) 있고 예약된 슬롯은 request가 부

사본의 마지막 블록 X^i_{K-1}을 액세스 한 후 다음 시간 주기에 request가 재배치되어 그 다음 블록인 주 사본 X^p_K을 액세스할 때만 사용한다고 가정하자. 즉, 예약 슬롯은 X^p_K을 읽을 때만 사용한다.

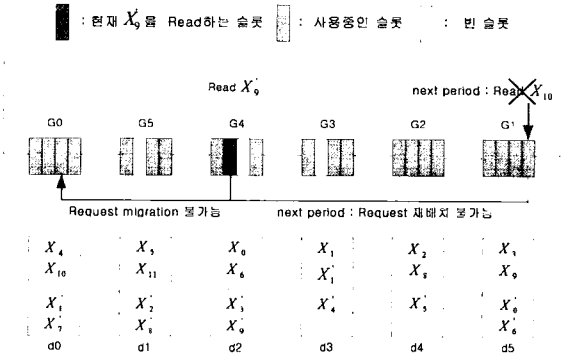

그림 2. 부분 복제 기술에서의 hiccup 발생

제안된 알고리즘은 버퍼를 사용하지 않고 hiccup 발생을 줄일 수 있다. 부 사본의 마지막 블록 X^i_{K-1}에 해당하는 그룹의 request가 X^i_{K-1}을 액세스하고 그 다음 시간 주기에 X^p_K을 읽기 위해 request가 X^p_K을 액세스할 수 있는 그룹에 재배치되려고 할 때 빈 슬롯이 없다면 이 때 예약된 슬롯을 사용하는 것이다. 제안된 알고리즘은 request가 재배치 될 그룹에 빈 슬롯이 없어도 request migration처럼 버퍼를 사용할 필요가 없다. 예를 들면, 그림 3에서와 같이 그룹 G_4가 부 사본의 마지막 블록 X^i_9에 도달할 때까지 그룹 G1에 빈 슬롯이 없다면 그 다음 시간 주기에 X_{10}을 읽기 위해서 예약된 슬롯을 이용한다.

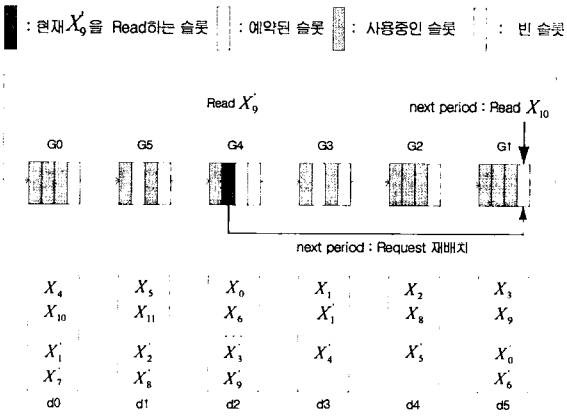

그림 3. 부분 복제 기술에서의 슬롯 예약 방법

이 슬롯 예약 알고리즘은 최대 처리율이 다소 줄어들지만 버퍼를 사용하지 않을 수도 있고 hiccup발생을 해결할 수 있다. 예를 들면, 모든 그룹에서 슬롯을 하나씩 예약해 놓으면 이 시스템의 최대 처리율은 d× (N-1)로 줄어들지만 버퍼를 사용하지 않을 수 있을 뿐만 아니라 hiccup이 발생하지 않는다. 모든 그룹의 슬롯을 하나씩 예약하지 않고 짝수 번째 혹은 홀수 번째 그룹에서 슬롯을 하나씩 예약해 놓는다면 시스템의 최대 처리율은 [d × N - $\lceil \frac{d}{2} \rceil$] 또는[d× N - $\lfloor \frac{d}{2} \rfloor$]가 된다. 반면에 모든 그룹의 슬롯을 하나씩 예약하는 것보다 hiccup이 발생할 확률은 커진다.

이 슬롯 예약 알고리즘은 버퍼를 가지고 request migration하는 것보다 최대 처리율이 다소 감소하기 때문에 소수의 사용자들이 접속하는 서버에 적합하다.

4. 결론

본 논문에서는 연속미디어에 있어서의 hiccup-free 디스플레이와 startup latency를 줄이기 위한 복제 기술에 대해서 소개하였다. 복제 기술 중에서 부분 복제에서 그룹의 비어있는 슬롯이 없을 경우, 임시 버퍼를 가지고 request migration하는 것이 불가능하여 hiccup이 발생할 수 있는 경우에 그룹의 슬롯을 예약함으로써 hiccup-free 디스플레이를 할 수 있는 알고리즘을 제안하였다. 향후 통계적 데이터를 이용한 시뮬레이션을 통해 가장 효율적인 예약 슬롯의 수를 결정하는 것과, 최대 처리율의 감소와 hiccup 문제 감소의 상관관계, 버퍼와 예약 슬롯의 관계에 대한 연구가 지속적으로 요구된다.

참고문헌

[1] 김완규, 박규석. VOD 시스템에서 클라이언트 버퍼를 위한 전송율 제어 알고리즘의 설계 및 분석. 1998년도 논문지 - 한국멀티미디어학회 1998 pp.67-78.

[2] 이경숙, 하숙정, 배인한. 주문형 비디오 시스템을 위한 예약 기반 동적 배칭 정책. 1998년도 추계학술발표논문집 - 한국멀티미디어학회 1998 pp.3-8.

[3] 이재경, 이경숙, 배인한. 주문형 비디오 서버를 위한 동적 디스크 스트라이핑과 데이터 배치 방법. 한국멀티미디어학회 - 1998년도 춘계학술발표논문집 1998 pp.10-15

[4] Seon Ho Kim. Replication Techniques to Minimize the Startup Latency of Continuous Media Servers. To appear in SCI 2001, July 2001.

[5] Seon Ho Kim. Bulk Prefetching with Deadline-Driven Scheduling to Minimize Startup Latency of Continuous Media Servers. To appear in IEEE ICME 2001, Aug. 2001.

[6] T.Ibaraki and N. Katoh. Resource Allocation Problems - Algorithmic Approaches. The MIT Press, 1988.

[7] Shahram Ghandeharizadeh, Seon Ho Kim, Weifeng Shi, Roger Zimmermann. On Minimizing Startup Latency in Scalable Continuous Media Servers. In the Proceedings of Multimedia Computing and Networking Conference, Feb. 1997.