

## 복제 웹 서버의 콘텐츠 동기화 알고리즘

김선호<sup>o</sup>, 고 훈, 신용태  
동덕여자대학교 정보학부<sup>o</sup>, 대진대학교 컴퓨터공학과, 숭실대학교 컴퓨터학과  
shkim98@dongduk.ac.kr<sup>o</sup>, skoh21@daejin.ac.kr, shin@computing.ac.kr

### Consistency Maintenance Algorithm of Replica Web Servers

Seonho Kim<sup>o</sup>, Hoon Ko, Yongtae Shin  
Dept. of Information and Science, Dongduk Women's University<sup>o</sup>  
Dept. of Computing and Engineering, Daejin University  
Dept. of Computing, Soongsil University

#### 요 약

인터넷상의 사용자와 대용량 콘텐츠의 급증에 따른 서비스 저하의 문제를 해결하기 위하여 웹 서버를 분산 설치하여 지역 서버로 하여금 클라이언트의 요청에 응답하도록 하는 콘텐츠 분산 네트워크(CDN)가 사용되고 있다. 그러나 원본 서버로부터 분산된 지역 서버에 콘텐츠를 복제하여 서비스하는 CDN 환경에서는 원본 서버와 복제 서버와의 콘텐츠 일관성을 유지하는 일이 중요한 문제이다.

그러므로 본 연구에서는 콘텐츠 특성에 따라 Adaptive TTL(Time to Live)을 적용하고 원본 서버와 복제 서버가 서로 협력하는 콘텐츠 동기화 알고리즘을 제안한다.

#### 1. 서론

콘텐츠 분산 네트워크에서 복제 서버들은 사용자와 원본(Origin) 웹 서버 사이에 위치하여 원본 서버의 자원을 저장해 두었다가 사용자의 요청에 대해 원본 서버를 대신하여 응답함으로써 네트워크 대역폭 사용과 원본 서버의 부하를 줄이고 사용자에게 대한 응답시간을 단축시킨다.

그러나 원본 서버를 거치지 않고 복제 서버로부터 자원을 전송하기 때문에 사용자는 최근에 갱신된 내용이 아닌 이전 내용을 전송 받을 수 있다. 즉 실제 원본 서버에 있는 콘텐츠와 복제 서버의 콘텐츠 사이에 동기화 문제가 생기는 것이다. 콘텐츠 동기화 정책은 자원들이 얼마나 자주 변경되는가에 따라 달라질 수 있는데 콘텐츠 동기화 정책에 캐쉬 일관성 유지 정책을 적용할 수 있다[1].

콘텐츠 분산 네트워크는 캐쉬와는 달리 원본 서버가 복제 서버들의 위치를 이미 알고 있고 미리 콘텐츠를 전송해 놓는 개념이기 때문에 콘텐츠 동기화를 위해 원본 서버에서 주기적으로 콘텐츠의 fresh 여부를 검사하여 복제 서버들에게 콘텐츠를 브로드캐스트 하는 것이 합리적이라 할 수 있다.

본 연구에서는 원본 웹 서버의 무효화 기법과 복제

서버의 콘텐츠 특성별 Adaptive TTL에 따라 프리패치 하는 기법을 동시에 적용하여 콘텐츠의 QoS를 보장하는 동기화 알고리즘을 제안한다.

#### 2. 관련연구

##### 2.1 TTL(Time to Live)

복제 서버에 있는 자원이 언제까지 유효할 것인지를 결정하기 위해 각 자원의 TTL 필드에 유효 시간(life time)을 지정하는 것이다. 이 유효 시간이 지나면 자원은 유효하지 않은 것으로 판단하고 사용자 요청 시 원본 서버로부터 다시 자원을 가져오게 된다. 이때, TTL을 너무 짧게 지정하게 되면 불필요하게 원본 서버로부터 다시 가져오는 일이 많아져 네트워크 오버헤드가 발생하고 TTL을 너무 길게 하면 유효하지 않은 자원을 사용자에게 전달할 가능성이 높아진다[3].

##### 2.2 Adaptive TTL

Last-Modified-Time(LMT)을 이용하여 자주 변경되는 자원일수록 자주 유효성 검사를 하고 자주 변경되지 않는 자원에 대해서는 유효성 검사를 자주 하지 않도록 하여 TTL의 단점을 보완한다[3].

##### 2.3 사용자 무효화(Invalidation) 메시지

IF\_MODIFIED\_SINCE(IMS) 헤더를 사용자의 요청이

있을 때마다 원본 서버에 보내어 유효성을 검사하고 변경 시 원본 서버로부터 다시 자원을 가져오는 방법으로 확실하게 일관성은 유지할 수 있지만 변경되지 않은 자원에 대해서도 IMS 메시지를 보내게 되기 때문에 불필요한 오버헤드가 발생할 수 있다[4].

**2.4 서버 무효화(Invalidation) 메시지**

원본 서버가 유효성을 검사하는 것으로 원본 서버의 자원이 변경된 경우 원본 서버의 복사 본을 가지고 있는 모든 서버에 변경을 알리는 것이다. 이것은 원본 서버의 복사 본을 가지고 있는 모든 서버에 관한 정보를 기록하고 있어야 한다[2].

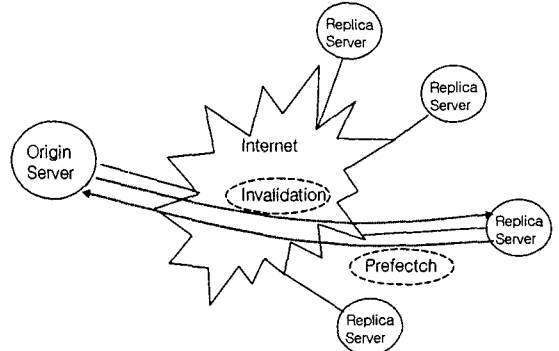


그림 1. 동기화 모델

**3. 콘텐츠 동기화 알고리즘**

연구에 의하면 웹상에서 사용자 요청의 67% 이상이 이미지 타입이며[5], 이미지 타입의 95% 이상이 평균 3.8개월 동안 변경되지 않는 것으로 나타났다[6]. 그러므로 용량이 크고 변경이 거의 없는 이미지 타입과 나머지 타입의 콘텐츠를 따로 구분하여 콘텐츠 특성에 따른 동기화 정책이 필요하다. 또한 요청이 있을 때 유효성을 검사하는 것보다는 미리 유효성을 검사하여 복제 서버에 push 하는 방법이 사용자의 요청에 빠른 응답을 할 수 있다.

이러한 특성을 고려하여 본 연구에서는 그림 1과 같이 원본 서버와 복제 서버가 협력하는 양방향 동기화 메커니즘을 제안한다. 원본 서버는 주기적으로 복제 서버에게 무효화 메시지(invalidation message)를 방송(broadcast)하여 갱신된 정보를 알린다. 복제 서버는 자주 변경되지 않는 객체와 잦은 변경이 일어나는 객체를 구분하고 사용자의 요청 빈도에 따라 각기 다른 객체 유효기간을 적용하여 객체 유효성 검사를 위한 메시지를 줄이고, 유효 기간이 지나면 원본 서버에게 콘텐츠를 요청하여 프리패치 한다. 이것은 클라이언트의 요구에 복제 서버가 즉각 반응 할 수 있도록 하고 콘텐츠 특성에 따라 TTL을 적용함으로써 전송량을 줄이고 대역폭을 절약한다.

프리패치(prefetch)는 사용자들이 자주 요구하는 데이터를 미리 원본 서버로부터 받아서 저장해 두는 기법이다[7]. 복제 서버는 프리패치 알고리즘을 이용하여 유효 기간이 만료된 콘텐츠를 프리패치하여 클라이언트의 요청 시에 즉시 응답한다.

각 웹 서버는 저장하고 있는 문서들에 대해 다음과 같은 메타 정보를 가지고 있으며 그림 2와 같이 표시할 수 있다.

- 유효 시간(Validated\_Time) : 문서의 유효성을 검사한 시간
- 수정 시간(Modified\_Time) : 문서가 원본 웹 서버에서 변경된 시간
- 현재 시간(Current\_Time) : 현재 시간
- 문서의 나이(Age) : 문서가 변경된 것을 확인한 시간부터 현재까지의 시간
- 지난 나이(Last\_Age) : 문서가 서버에서 변경된 시간부터 변경된 사항을 확인할 때까지의 시간

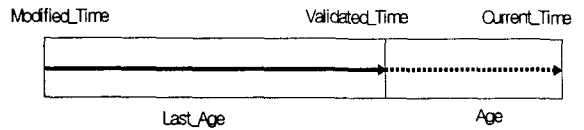


그림 2. 메타 정보의 표현

프리패치 시에는 문서가 유효한지와 문서의 참조 가능성이 높은지를 고려하여야 한다. 문서의 유효성은  $Current\_Time - Validated\_Time$  값이 작을수록,  $Validated\_Time - Modified\_Time$  값이 클수록 높다고 할 수 있다. 문서의 참조 가능성은 최근에 참조된 문서일수록 앞으로 참조될 가능성이 높다고 볼 수 있으므로 문서의 유효성을 검사한 시간(Validated\_Time)이 최근일수록 참조 가능성이 높다고 할 수 있다.

그러므로 다음과 같은 식 (1)에 의해 프리패치를 결정할 수 있다.

$$Prefetch\_Time = Validated\_Time * \frac{Age}{Last\ Age} \quad (1)$$

다음 그림 3의 알고리즘과 같이 원본 서버에서는 일정한 주기가 되었을 때(*Fresh\_Check\_Time*), 트래픽이 한가한 시간대에(*Low\_Traffic\_Time*) 콘텐츠의 변경이 있었는지 점검하여 변경된 정보를 복제 서버들에게 일괄 브로드캐스트한다. HTTP 1.1에서는 원본 서버가 put, delete, post 요청을 이용하여 복제 서버에 복사된 객체를 무효화 할 수 있다.

**Invalidation Algorithm of Origin Server**

```

if !(Fresh_Check_Time and Low_Traffic_Time)
    exit;
if (Modified_Time < Validated_Time)
    broadcasting update message to replicas
else
    broadcasting <EMPTY>;
    
```

그림 3. Invalidation Algorithm

**Refresh Algorithm of Replica Servers**

```

if (Client_Max_Age < Age) Stale ;
if (Expire < Current_Time) Stale ;
if (Prefetch_Time < Allowed_Time )
    Fresh ;
else
    if (Prefetch_Time >= Allowed_Time)
        and (Last_Age > Image_Valid_Time)
            Stale ;
    if (Age < MIN) Fresh ;
    o.w Stale ;
    
```

그림 4. Refresh Algorithm

복제 서버에서는 스쿼드(Squid) 프락시에서 사용되고 있는 Adaptive TTL 메커니즘을 수정하여 그림 4와 같이 *Prefetch\_Time*이 지정한 시간(*Allowed\_Time*)내에 있는지, 이미지 타입의 TTL(*Image\_Valid\_Time*)이 초과 했는지 등을 점검하여 콘텐츠의 유효성이 *Stale* 하면 내용을 갱신한다.

양방향으로 메시지가 전달되지만 서비스되는 콘텐츠의 특성을 고려하여 무효화 주기와 유효 기간을 설정할 경우 기존에 비해 메시지 전송 횟수를 줄일 수 있으며

미리 fresh 여부를 점검해 두기 때문에 클라이언트의 요청이 있을 때 바로 응답할 수 있어 지연을 줄일 수 있다.

**4. 결론 및 향후 연구 계획**

본 연구에서는 콘텐츠 특성을 고려하고 원본 서버와 복제 서버가 협력하여 콘텐츠의 동기화를 유지하는 알고리즘을 제안하였다. 향후 제안한 알고리즘을 적용하여 콘텐츠의 히트율과 미스율을 실험할 계획이다.

**5. 참고문헌**

- [1] Z. Fei., A Novel Approach to Managing Consistency in Content Distribution Networks, *In Proceedings of the 6th Workshop on Web Caching and Content Distribution*, 2001.
- [2] P. Ceo and C. Liu, Maintaining Strong Cache Consistency in the WWW, *IEEE Transactions on Computers*, 47(4) pp. 445-457, 1998.
- [3] J. Gwertzman and M. Seltzer, World Wide Web Cache Consistency, *In Proceedings of USENIX Technical Conference*, pp. 141-151, 1996.
- [4] D. Li, P. Cao, and M. Dahlin, WCIP: Web Cache Invalidation Protocol, *IEFT Internet draft*, 2000.
- [5] S. Gribble and E .A. Brewer, System Design Issues for Internet Middleware Services: Deductions from a Large Client Trace, *In Proceedings of the USENIX Symposium on Internet Technologies and Systems*, 1997.
- [6] F. Douglis, A. Feldmann, B. Krishnamurthy, and J. Mogul, Rate of Change and Othermetrics : a Live Study of the World Wide Web. *In Proceedings of USENIX Symposium on Internet Technologies and Systems*, 1997.
- [7] V. N. Padmanabhan, J. C. Mogul, Using Predictive Prefetching to Improve WWW Latency, *ACM SIGCOMM Computer Communication Review*, July 1996.