예측된 선호도 기반 게으른 캐싱 전략

박 철† · 유 해 영††

요 약

본 논문에서는 파일이 요청된 순간에는 파일의 선호도만을 조사하고, 일정 시간이 흐른 후에 선호도가 높을 것으로 예상되는 파일들을 일괄적으로 캐싱하는 새로운 캐싱 전략을 소개한다. 예측된 선호도 기반 게으른 캐싱 전략(Forecasted Popularity Based Lazy Caching Strategy)은 웹 파일의 선호도를 지수평활법을 사용하여 예측하여 기존의 캐싱 전략보다 높은 캐시 적중률을 보여준다. 국내외 5개 웹 서버로부터 수집한 로그 파일을 대상으로 실험한 결과에 의하면, 선호도의 예측이 정확할수록 높은 캐시 적중률을 나타낸다. 이는 웹 파일의 선호도 예측 기법에 대한 연구를 통해 캐시의 성능을 향상시킬 수 있음을 보여준다.

Forecasted Popularity Based Lazy Caching Strategy

Chel Park† · Hae-Young Yoo††

ABSTRACT

In this paper, we propose a new caching strategy for web servers. The proposed strategy collects only the statistics of the requested file, for example the popularity, when a request arrives. At a point of time, only files with higher forecasted popularity are cached all together. Forecasted popularity based lazy caching (FPLC) strategy uses exponential smoothing method for forecast popularity of web files. And, FPLC strategy shows that the cache hit ratio and the cache transfer ratio are better than those produced by other caching strategy. Furthermore, the experiment that is performed with real log files built from web servers shows our study on forecast method for popularity of web files improves cache efficiency.

키워드 : 웹캐싱전략(Web Caching Strategy), 지수평활법(Exponential Smoothing), PLC, FPLC, 예측된 선호도 기반 게으른 캐싱 전략 (Forecasted Popularity Based Lazy Caching Strategy)

1. 서 론

웹의 사용범위는 급속한 속도로 증가되어 왔다. 학교, 관공서, 대기업 위주로 보급되던 인터넷 사용이 대부분의 사무실, 가정으로 보급되었고, 초고속 인터넷이 800만 가구 이상에 보급되면서 국내 인터넷 이용에서 통신속도 향상에 많은 발전이 이루어 졌다[12]. 사용자의 증가는 전송되는 데이터의 양을 증가시키게 되어서 다른 사용자를 위한 통신지연을 증가시키는 원인이 되었다. 사용자의 통신지연을 감소시키기 위해서는 빠른 통신망의 확충은 기본적으로 요구되는 사항이며, 자주 요청되는 객체의 복사본을 그 사용자와 가까운 곳에 저장하기 위해 프록시 서버를 사용하거나, 웹 서버의 성능을 개선하여 웹 서버가 요청 받은 파일을 신속하게 서비스 하도록 하는 연구 등이 다양한 관점에서 연구되고 있다[2, 7, 8].

웹 서버에서 캐싱 전략을 사용하는 것은 서버가 요청을 처리하는 시간을 줄임으로써 사용자가 파일을 요청한 후 응답을 받을 때까지의 시간을 줄여주어 사용자 통신지연을 줄여주는 효과를 주게 됨으로 이에 대한 다양한 캐싱 전략들이 연구되었다[1, 3-5].

선호도 기반 게으른 캐싱(Popularity Based Lazy Caching, PLC) 전략은 기존의 다른 캐싱 전략들과 차별화 된 특징을 가지고 있다[9]. 기존의 캐싱 전략들은 공통적으로 매번 요청이 발생하는 순간마다 파일의 크기나 캐시의 크기, 파일들의 선호도, 파일들 사이의 관계, 시간적 국지성 등의 기준에 의해서 "이 파일을 캐싱하는 것이 유리한가? 어떤 파일을 캐시에서 제거하는 것이 유리한가?"를 판단한다는 점이다. 선호도 기반 게으른 캐싱 전략은 웹에서 요청되는 파일의 선호도의 변화가 충분히 느리기 때문에 매번 캐싱 여부를 판단하지 않고 하루 또는 그 이상의 주기로 캐싱하는 것으로도 기존의 캐싱 전략 이상의 성능을 보인다.

그러나 이러한 연구들에서 선호도를 과거에서 현재까지의 일정 구간에서 요청된 요청빈도로 사용하고 있어서 실

† 종신회원 : 단국대학교 정보컴퓨터학부 교수
†† 정 회 원 : 단국대학교 정보컴퓨터학부 교수
논문접수 : 2003년 3월 11일, 심사완료 : 2003년 6월 12일

제 캐싱이 이루어지는 시점에서의 선호도와는 차이가 있게 된다.

이에 본 연구에서는 첫째, 선호도의 변화경향을 비교적 간단한 예측 기법인 지수평활법을 이용하여 선호도를 예측하는 방안을 제안하고, 적절한 평활 상수를 구하기 위하여 웹 서버의 로그 파일을 분석하였다. 둘째, 예측된 선호도가 적정한 의미를 가지고 있는 가를 검증하였다. 이를 위하여 PLC 전략을 수정한 예측된 선호도 기반 게으른 캐싱(FPLC) 전략을 제안하고 선호도 예측의 정확성이 캐싱 성능에 미치는 영향을 실험하였다.

2. 웹 서비스 성능 향상 기법

2.1 웹 서비스 성능 향상 기법들

웹 서비스의 성능 향상을 위한 방법 중 캐싱 전략은 캐싱이 이루어지는 위치에 따라 클라이언트(웹 브라우저)에서의 캐싱 전략, 프록시 서버에서의 캐싱 전략, 웹 서버에서의 캐싱 전략으로 구분된다. 클라이언트에서의 캐싱 전략은 디스크 또는 캐시 메모리에 적재되어 있는 캐싱된 자료를 검색하는 속도와 네트워크와의 속도 차이를 이용하여 웹 서비스의 검색 시간을 줄이는데 목적이 있으며, 프록시 서버에서의 캐싱 전략은 로컬 네트워크와 외부 네트워크와의 속도 차이를 이용하여 검색 시간을 줄인다. 반면에 웹 서버에서의 캐싱 전략은 웹 서버가 디스크에 있는 파일을 메모리에 캐싱하여 웹 서버가 클라이언트의 요청에 응답하는 요청 처리 시간을 줄이는데 목적이 있다.

2.2 메인 메모리 캐싱

웹 서버에서의 캐싱 전략에 대한 대표적인 연구인 메인 메모리 캐싱은 Markatos가 제안한 방식으로 자주 요청되는 웹 문서를 웹 서버 프로그램의 주소공간에 캐싱하는 전략이 있다[3]. 기존의 웹 서버에 존재하는 파일 시스템이 이미 디스크 캐시를 가지고 있지만 기존의 디스크 캐시들은 웹 작업부하에 최적화되어 있지 못하다. 첫째, 캐싱의 단위가 다르다, 전통적인 파일 시스템의 캐싱 전략은 각각의 디스크 블록에 적용되지만 웹 서버는 전체 파일단위로 필요로 하기 때문에 파일 단위의 전략이 필요하다. 둘째, 불필요하게 운영체제가 개입된다. 셋째, 웹 문서는 대부분 읽기 전용으로 사용된다. 메인 메모리 캐싱은 캐시를 웹 서버의 주소공간에 두어 웹 작업부하에 적절하게 대응할 수 잇는 캐싱 전략을 제안한 것이다.

Adaptive-level 메모리 캐싱은 메인 메모리 캐싱을 개선하여 파일 크기를 threshold를 기준으로 하여 기준보다 크기가 작은 파일은 전체 문서를 캐싱하고 크기가 큰 파일들은 파일의 첫 번째 조각을 캐시에 저장한다[1]. 이러한 전략은 크기가 큰 파일의 경우 웹 서버는 첫 번째 조각을 클라이언트에 전송하면서 동시에 파일의 나머지 부분을 디스크로부터 읽어 들일 수 있어서 클라이언트에게 데이터를 전송하기 전에 소요되는 요청 처리 시간을 줄일 수 있다. 현재의 웹 서버의 작업부하는 크기가 작은 파일들이 높은 선호도를 가지고 있어서 현재는 작업부하에서는 현저한 성능향상을 기대할 수 없지만, 웹에서 사용되는 파일들의 크기가 점차 커지고 있고 큰 파일들에 대한 선호도가 높아질 경우 adaptive-level 메모리 캐싱이 확연한 성능 향상을 보인다.

3. 예측된 선호도 기반 게으른 캐싱(FPLC) 전략

FPLC 전략은 유해영이 제안한 선호도 기반 게으른 캐싱(PLC) 전략[9]에 선호도 예측 기법을 도입하여 캐싱의 효율을 높인 캐싱 전략이다.

3.1 선호도 기반 게으른 캐싱 전략(PLC)

서버가 보유한 파일들의 선호도는 시간이 흐름에 따라 꾸준히 변하기는 하더라도, 특별한 경우를 제외하면, 파일이 요청되는 순간 순간마다 이 파일을 캐싱하는 것이 이로운지 혹은 별 이득이 없는지를 판단하고 이에 따라 파일을 즉시 캐싱 여부를 판단하야 할 정도로 급격하게 변하지는 않는다[10]. 이러한 특징을 캐싱 전략에 적용하기 위해서, 과거의 일정기간 동안 선호도가 높았던 파일들에 대한 요청이 가까운 미래에도 많이 요청되어질 가능성이 높아질 것이므로, 선호도 기반 게으른 캐싱(PLC) 전략[9]에서는 웹 서버에서 캐싱 전략을 linear knapsack 문제로 모델링하고 그 결과를 이용하여 서버의 성능 향상에 기여가 예상되는 선호도가 높은 파일들을 추출하고 이들을 일괄로 캐시에 캐싱한다.

임의의 캐싱 전략 A의 i 구간에서의 성능은 C(i, A)로 정의할 수 있다.

$$C(i,\ A) = \sum_{f}^{all\ file} C(i,\ f,\ A) \qquad (1)$$

$C(i,\ f,\ A) =$ *파일 f를 캐시에서 서비스한 경우의 수* $/ A(i)$
$A(i) =$ *i 구간의 총요청수*

C(i, f, A)를 다시 표현하면 식 (2)가 된다.

$$C(i,f,A) = \frac{\text{파일 } f\text{를 캐시에서 서비스한 경우의 수}}{A(i)} \times \frac{A(i,\ f)}{A(i,\ f)} = \frac{\text{파일 } f\text{를 캐시에서 서비스한 경우의 수}}{A(i,\ f)} \times \frac{A(i,\ f)}{A(i)} \qquad (2)$$

R(i, f, A)와 p(i, f)를 식 (3)으로 정의하면 C(i, f, A)는 R(i, f, A)와 p(i, f)의 곱으로 표현되어 진다.

$$C(i, f, A) = R(i, f, A) \times p(i, f)$$

$$R(i, f, A) = \frac{\text{파일 } f\text{를 캐시에서 서비스한 경우의 수}}{A(i, f)} \quad (3)$$

$$p(i, f) = \frac{A(i, f)}{A(i)}$$

R(i, f, A)는 A라는 캐싱 전략을 사용했을 때, 파일 f에 대한 총 요청 중에서 캐시에서 서비스된 비율을 의미한다. 예를 들어 R(i, f, A)=1 이라면 A 캐싱 전략을 사용하는 경우 f라는 파일은 요청되었을 때 항상 캐시에서 서비스 됨을 의미하며, R(i, f, A)=0이면 파일 f가 요청될 때마다 디스크에서 읽어서 서비스한다는 것을 의미한다.

p(i, f)는 전체 요청 중에서 파일 f가 요청된 비율로써 파일 f의 i구간에서의 선호도로 정의 된다. 따라서 p(i, f)는 캐싱 전략과는 독립적이며, 사용자들의 선호에 의해서 결정되는 각 파일의 특성임을 알 수 있다.

A캐싱 전략의 성능을 높이는 문제는 식 (4)로 정의된다.

$$\begin{aligned} \text{캐시성능} &= maximize\ C(i, A) \\ &= maximize \sum_{f}^{all\ file} C(i, f, A) \\ &= maximize \sum_{f}^{all\ file} R(i, f, A) \times p(i, f) \end{aligned} \quad (4)$$

이 문제는 pure binary integer programming에 속한다. pure binary integer programming은 linear programming relaxation이라고 불리는 linear knapsack 문제로 변환하여 해결하면 간단해 진다[6]. 이 문제의 최적해는 critical efficiency e를 식 (5)와 같이 정의하고 e가 큰 값부터 순서대로 캐시에 넣어주면 된다[5]. critical efficiency를 이용하는 유해영[9]이 제안한 PLC 전략의 알고리즘이 (그림 1)이다.

$$\text{critical efficiency } e = \frac{p(i, f)}{S(f)},\ S(f) = \text{파일 } f\text{의 크기} \quad (5)$$

$$p(i, f) = \text{가장 최근구간의 파일 } f\text{의 선호도}$$

```
Execute the following whenever a request arrives
  let f be the file in the request
  update critical efficiency of f
  if f is in the cache
      transfer the content reading from cache
  else
      read f from disk and transfer it to client
  end if
  if the current time is the time that update cache then
      get L that is the list of files with higher critical efficiency
      Destroy and build new cache with L
      Reset measure for all files
  end if
```

(그림 1) PLC 전략 알고리즘

3.2 예측된 선호도 기반 게으른 캐싱(FPLC) 전략

식 (4)를 보면 캐시의 성능은 R(i, f, A)와 p(i, f)의 두 가지 항목에 의하여 결정된다. 따라서 PLC 전략은 과거부터 캐시가 갱신되는 시점까지의 파일의 선호도를 이용하여 각 파일의 critical efficiency를 계산하여 캐시를 갱신한 후 다음 캐시의 갱신 주기가 될 때까지 사용한다. 따라서 실제 캐시의 성능은 과거의 파일의 선호도에 의해 결정되는 것이 아니고 캐시가 갱신된 그 다음 구간에서의 실제적인 선호도에 의해서 결정되게 된다.

PLC 전략이 각 파일의 선호도가 급격한 변화가 없다는 점을 이용하고 있지만, 급격한 변화는 없다 하더라도 작은 규모의 선호도의 변화는 실제로 존재한다. PLC 전략의 최적해가 정확한 의미를 가지기 위해서는 캐시가 갱신된 이후의 구간에서 각 파일의 선호도가 정확해야 한다. 그러나 아직 요청되지 않은 구간에서의 선호도를 미리 알 수는 없기 때문에 예측 기법을 사용하여야 한다. 이에 본 논문에서는 PLC 전략을 예측 기법을 이용하여 예측된 선호도를 사용하도록 수정한 FPLC 전략을 소개한다.

FPLC 전략에서는 예측 기법으로 간단한 지수평활법을 사용한다. 미래의 선호도를 예측하기 위한 방법으로는 본 논문에서 이용한 지수평활법 이외에 자기회귀 모델(AR 모델), 이동평균 모델(MA 모델), 자기회귀이동평균 모델(ARMA 모델), ARIMA 모델, Box-Jenkins 모델 등의 보다 효율적인 시계열 모델들을 사용할 수 있다[11]. 본 연구에서는 선호도를 예측하는 최적의 모델을 발견하는데 목적이 있는 것이 아니고 파일의 장기적인 선호도를 예측하는 것이 캐시 성능에 긍정적인 영향을 미치는 것을 보이는데 목적이 있기 때문에 예측기법으로 간단한 지수평활법을 사용하였다.

지수평활법은 전기의 실측치와 전기의 예측치를 일정한 비율로 가중 평균하여 당기 예측 치로 사용하는 예측 방법으로 식 (6)을 사용하여 미래의 수요를 예측한다.

$$\begin{aligned} \text{당기 예측치} &= (\text{전기 실측치}) + (1 - \alpha) \times (\text{전기예측치}) \\ \alpha &= \text{평활상수},\ 0 < \alpha < 1 \end{aligned} \quad (6)$$

식 (6)을 식 (7)과 같이 변형하면 지수평활법은 전기의 예측오차를 α의 비율로 더하여 차기의 예측치를 수정해나가는 방식이다.

$$\begin{aligned} \text{당기 예측치} &= \text{전기 실측치} + \alpha \times (\text{전기실측치} - \text{전기예측치}) \\ &= \text{전기 예측치} + \alpha \times (\text{전기 예측오차}) \end{aligned} \quad (7)$$

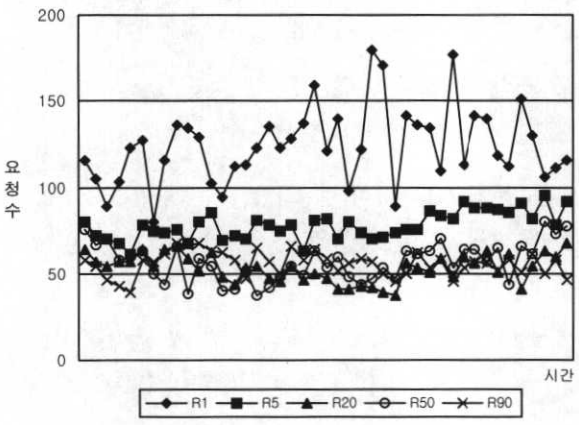
위의 수식에서 α는 평활상수로 부르며 지수평활법을 실제 예측에 사용할 때 가장 중요한 문제는 바로 적절한 평활상수를 결정하는 것이다. 평활상수 α는 예측치가 과거의 예측오차에 반응하는 정도를 결정한다. α가 클수록 예측치는 실제치를 더 가까이 따라가며 α값이 작으면 작을수록

평활의 효과는 더 커지게 된다.

최상의 예측 성능을 보이는 평활상수는 예측치와 실측지의 오차제곱합이 최소가 되는 값을 사용하면 된다. 본 논문에서는 예측의 최상의 평활상수의 위치를 시각적으로 판단할 수 있는 그레이-레벨 밴드를 사용한다. (그림 2)의 가로 방향으로 한 줄의 이미지를 그레이-레벨 밴드라고 부른다. (그림 2)는 한 개의 파일에 대하여 가장 왼쪽이 평활상수가 0.99인 경우이고 오른쪽으로 이동하면서 0.01인 감소해서 가장 오른쪽이 평활상수가 0.01인 경우의 오차제곱합의 크기를 회색의 농도로 표현한다. 색의 농도가 검은색인 경우는 해당 평활상수를 사용했을 때 오차제곱합이 큼을 의미하고 색의 농도가 하얀색에 가까울수록 오차 제곱합이 작음을 의미한다. 따라서 (그림 2)의 띠에서 흰색인 위치가 최상의 예측 성능을 보이는 평활상수가 된다. (그림 2)의 경우 예측 성능이 높은 구간을 강조하기 위해 예측 성능이 1등인 것을 흰색으로 해서 10등까지만 회색으로 표현하였다.

(그림 2) 그레이-레벨 밴드 설명

파일의 선호도를 지수평활법으로 예측하려면 각 파일별로 적절한 평활상수를 결정해 주어야만 한다. (그림 3)는 HP01 서버에서 선호도가 전체구간에서 1등, 5등, 20등, 50등, 90등인 5개의 파일(각각 R1, R5, R20, R50, R90)의 구간별 요청수를 시간 순으로 나열한 그래프이다. 그래프를 보면 각 파일의 선호도의 변화되는 경향이 서로 다르다는 것을 알 수 있다. 따라서 각각의 파일마다 최상의 예측 성능을 보이는 평활상수가 서로 다른 값을 가지며 존재한다.



(그림 3) 파일의 선호도의 차이

그러나 캐싱전략에서 각 파일의 선호도를 예측하기 위하여 파일의 수 만큼의 평활상수를 기억하고 있는 것은 적절하지 못하다. (그림 4)은 HP01 서버에서 선호도가 높은 파일들에 대하여 최상의 평활상수의 위치를 보여주는 그레이-레벨 이미지 이다. 그림에서 위쪽의 가로줄이 선호도가 높은 파일에 대한 그레이-레벨 밴드이고 아래로 내려갈수록 선호도가 2등, 3등의 순으로 각 파일들의 그레이-레벨 밴드를 200등까지를 쌓아놓은 것이다. (그림 4)를 보면 최상의 예측성능을 보이는 흰색 점들이 세로방향으로 띠를 이루고 있다. 이러한 띠는 동일한 웹 서버에 존재하는 파일들은 각각 다른 평활상수를 사용하는 대신에 웹 서버 단위로 한 개의 평활상수를 사용하여도 비슷한 예측 성능을 예상할 수 있다.

(그림 4) HP01 서버의 그레이-레벨 이미지

본 논문에서는 예측된 선호도를 이용하여 critical efficiency를 계산하고 이 값이 높은 파일들을 캐시에 넣는 캐싱 전략을 예측한 선호도 기반 게으른 캐싱(FPLC : Forecasted Popularity Based Lazy Caching) 전략으로 부르기로 한다. FPLC 전략에서 웹 서버의 각 파일의 선호도를 동일한 평활상수를 사용한 지수평활법으로 예측한다. 이렇게 수정된 FPLC 전략의 알고리즘은 (그림 5)와 같다.

```
Execute the following whenever a request arrives
    let f be the file in the request
    update popularity of f
     if f is in the cache then
            transfer the content reading from cache
    else
            read f from disk and transfer it to client
    check that the current time is the time to update cache
     if it is time to update cache then
            Get L that is the list of files with forecast popularity
            Destroy and build new cache with L
            Reset measure for all files.
```

(그림 5) FPLC 전략 알고리즘

4. 성능 평가

FPLC 전략이 PLC에 비해 높은 캐시 성능을 보이고 선호도의 예측이 정확함을 보이기 위하여 다음의 두 가지 실험을 하였다. 첫 번째 실험에서는 그레이-레벨 이미지를 사용하여 동일한 서버 내에 존재하는 파일들은 대부분 비슷한 평활상수에서 최상의 예측 성능을 가짐을 보여 FPLC 전략

에서 각 파일의 선호도 예측에 서버별로 유일한 평활상수를 사용할 수 있음을 보인다. 두 번째 실험에서는 FPLC 전략을 웹 서버의 로그 데이터를 이용하여 평활상수를 변화시키면서 시뮬레이션 한다. 이 실험에서는 적절한 평활상수를 선택하는 것이 즉, 선호도에 대한 예측이 정확할수록 FPLC 전략의 성능이 향상됨을 보인다.

4.1 실험 대상 로그 데이터

실험에는 2000년 7월 22일부터 8월 4일까지 2주(14일)동안, 국내의 대표적인 교육정보 포털 사이트인 에듀넷(www.edunet4u.net)을 운영중인 KERIS(한국교육학술정보원)의 3개 웹 서버의 각각의 로그 파일을 일자별로 수집하여 분석에 사용하였다.

이들 HP01, HP05, EDU01이라고 이름을 부여한 서버들의 로그 데이터의 수집기간과 컨텐츠, 총 접근 수, 총 전송량 등을 <표 1>에 나타내었다.

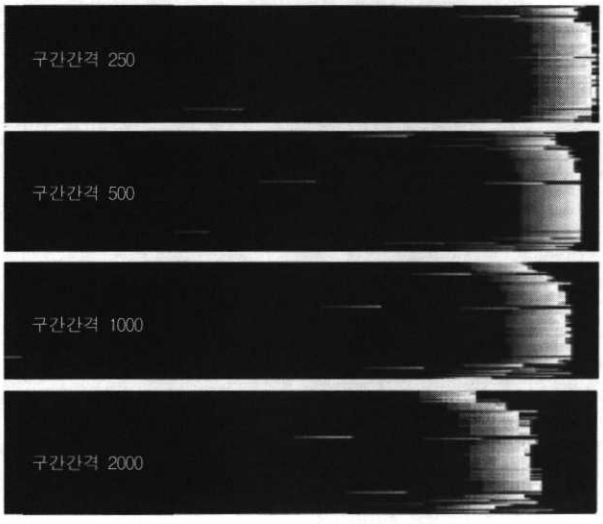
〈표 1〉 실험에 사용된 로그 데이터

	HP01	HP05	EDU01
수집기간	2000년 7월 22일 ~ 2000년 8월 4일(2주간)		
컨 텐 츠	평생교육	사이버 학습교재	사용자 인증 및 포탈
교육 대상	학생, 일반인	초, 중학생	사용자 전체
특기 사항	교재내용 제외	교재, 데이터	
총 접근 수	12,233,842	18,717,312	119,573,369
총 전송량	187,087MBytes	276,029MBytes	576,365,605,969MBytes

* 총 접근 수 : 수집 기간 중에 웹 서버에 접근한 총 횟수
* 총 전송량 : 수집 기간 중에 웹 서버에서 클라이언트에 전송된 데이터의 총량

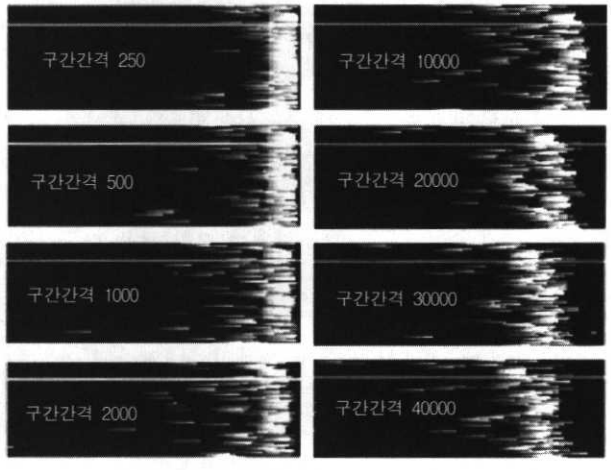
4.2 평활상수의 선택

각각의 실험대상 웹 서버의 로그를 이용하여 선호도를 측정하는 구간을 250~40000까지 다양하게 변화시키면서 생성한 그레이-레벨 이미지가 (그림 6), (그림 7), (그림 8)이다. 각각의 그림을 살펴보면 구간 간격이 적은 구간에서는 세로 방향으로 보이는 흰색 띠가 거의 직하향으로 발견되어진다. 이것은 최상의 예측 성능을 보이는 각 파일의 평활상수가 비슷한 값을 가짐을 의미한다. 그러나 구간 간격이 넓어질수록 점점 흩어지고 있으며 HP05 서버는 다른 서버들에 비하여 흩어짐의 정도가 심하여, HP05 서버는 각 파일의 선호도 특성의 다른 정도가 다른 서버보다 높음을 알 수 있다. 이는 HP05 서버가 초중고의 사이버학습 교재를 동시에 포함하고 있어 사이트에 접속하는 대상과 그들이 사용하는 컨텐츠가 밀접하게 연관되어 있기 때문으로 예상할 수 있다.



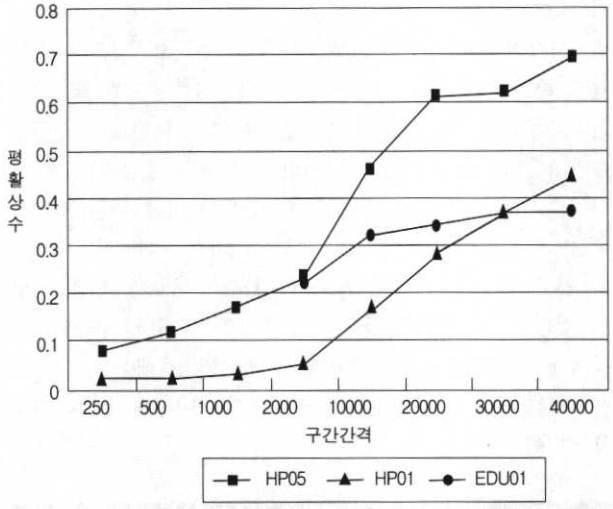
(그림 6) HP01 서버의 그레이-레벨 이미지

(그림 7) EDU01 서버의 그레이-레벨 이미지



(그림 8) HP05 서버의 그레이-레벨 이미지

(그림 9)는 각각의 파일에 따로 따로 평활상수를 적용하지 않고 각 서버별로 유일한 평활상수를 적용했을 때 구간별로 어느 평활상수가 최상의 예측 성능을 보이는지를 보여준다. 각 서버마다 평활상수는 다르지만 구간 간격이 넓어질수록 평활상수가 큰 값을 가지게 됨을 알 수 있다.



(그림 9) 각 서버에서 구간간격별 평활상수의 변화에 따른 예측의 정확도

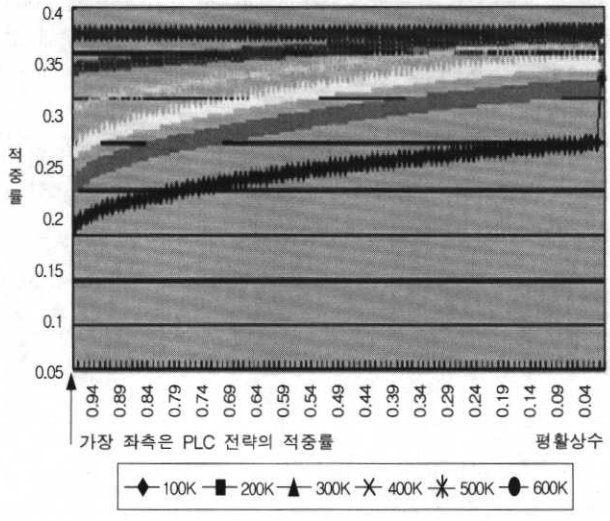
4.3 FPLC 성능 평가

첫 번째 실험에서는 지수평활법을 사용할 때 적절한 평활상수가 얼마인지를 조사하였다. 그러나 이것만으로는 예측된 선호도가 얼마나 정확한지 그리고 그 선호도가 얼마나 유용할지를 확인할 수 없다. 그래서 두 번째 실험에서는 예측된 선호도를 캐싱에 사용하는 경우 캐시 성능의 개선을 시뮬레이션을 통해 확인하였다. 시뮬레이션은 FPLC 전략을 사용하여 실험 대상 웹 서버들에 대하여 캐싱의 갱신 주기를 다양하게 변화시키면서 여러 캐시의 크기를 변화시키면서 평활상수에 따른 캐시 성능을 적중률을 사용하여 측정하였다.

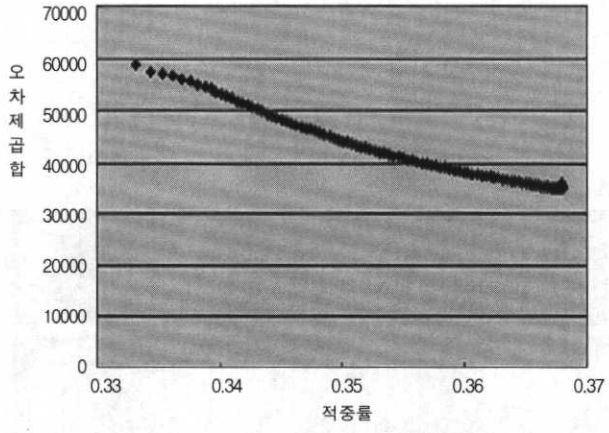
(그림 10)은 HP01 서버에서 캐시를 갱신하는 구간간격을 250으로 했을 때 측정된 적중률이다. 캐시 크기가 적은 경우 적중률은 평활상수의 영향을 많이 받으며 캐시의 크기가 커질수록 상대적으로 영향이 적음을 보여준다(단, 그래프에서 제일 왼쪽의 점은 선호도에 예측 기법을 적용하지 않는 PLC 전략의 적중률이고, 제일 오른쪽 점은 다음 구간의 값을 정확히 예측한 경우에 얻을 수 있는 최적의 적중률을 의미한다).

FPLC 전략을 사용했을 때 높은 캐시 적중률을 보이는 평활상수와 지수평활법을 사용했을때 오차제곱합이 최소가 되는 평활상수가 일치하는지 확인하기 위해여 캐시 적중률과 오차제곱합의 상관관계를 측정하였다. (그림 11)은 HP01 서버에서 두 데이터의 상관도이고 <표 2>는 HP01 서버에서 두 데이터의 상관계수를 구간간격별로 측정한 표이다. 표를 보면 두 데이터는 구간간격 10000이하에서는 0.9이상의 높은 음의 상관관계를 나타나고 있어서 오차제곱합이 작을수록, 즉 지수평활법을 사용한 예측 성능이 높을수록 캐시의 적중률이 증가함을 보여준다.

hp01 구간간격 250					
평활상수	100K	200K	300K	400K	500K
0.99	0.160509	0.203382	0.246030	0.290818	0.333188
0.98	0.165140	0.209510	0.250158	0.293208	0.334318
0.97	0.169233	0.213890	0.253454	0.295245	0.335261
0.96	0.172457	0.218095	0.256895	0.297324	0.336183
0.95	0.175493	0.221089	0.259508	0.298814	0.336881
0.94	0.177629	0.223748	0.262217	0.300490	0.337561
0.93	0.179721	0.226413	0.264709	0.301978	0.338211



(그림 10) HP01서버의 구간간격 250에서 평활상수 변화에 따른 적중률의 변화



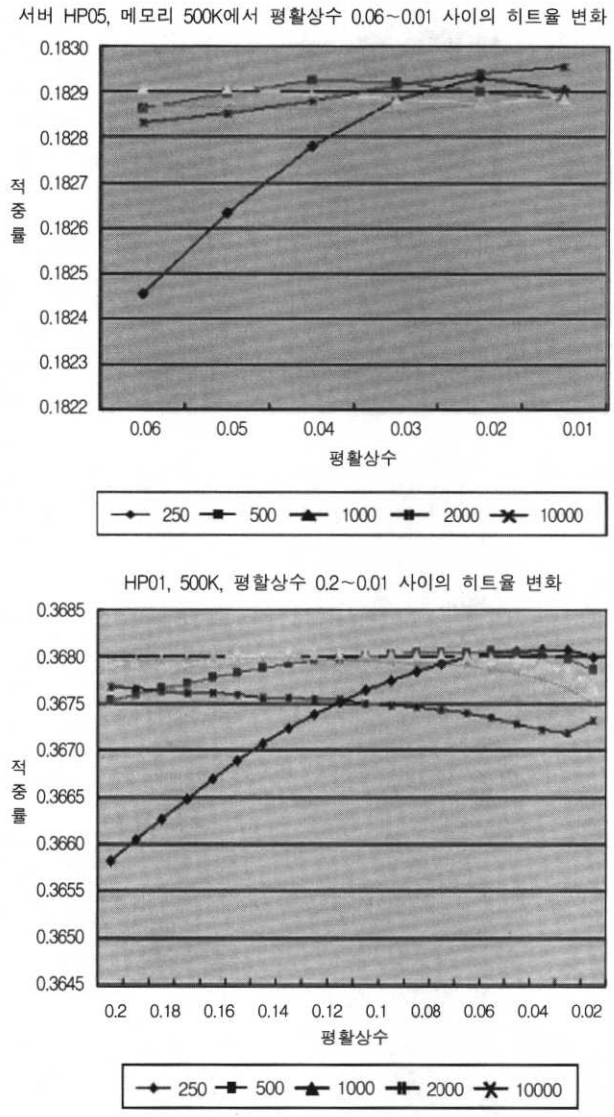
(그림 11) HP01 서버의 FPLC 적중률-오차제곱합 상관도

〈표 2〉 HP01 서버의 FPLC 적중률-오차제곱합 상관계수

캐시크기 : 500K							
250	500	1000	2000	10000	20000	30000	40000
-0.988	-0.966	-0.955	-0.918	-0.918	-0.855	-0.672	-0.569

(그림 12)는 (그림 10)의 평활상수의 변화에 따른 적중률의 변화 그래프에서 적중률이 정점을 이루는 부분을 확대한 그림

이다. 그림에서 각 곡선은 평활상수가 변화함에 따라서 부드러운 곡선을 그리면 적중률이 상승하고 있으며 우측의 어떤 지점을 지나면서 적중률이 감소하는 공통적인 패턴을 보여준다. 이러한 곡선은 캐시의 적중률이 어떠한 평활상수를 선택하는가에 따라서, 즉 선호도에 대한 예측이 정확해 질수록 FPLC 전략의 캐시 적중률이 민감하게 변화하는 것을 보여준다.



(그림 12) 평활상수 변화에 따른 적중률의 변화

5. 결 론

본 연구에서는 웹 파일의 선호도는 장기간에 걸쳐서 지속적으로 유지된다는 독특한 특성을 이용한 PLC 전략을 수정하여 다음 주기동안의 선호도를 예측기법을 통하여 보다 정확하게 예측하여 캐시 성능을 높이는 예측된 선호도 기반 게으른 캐싱 전략을 제시하였다.

웹 파일의 선호도 예측에 비교적 간단한 지수평활법을 사용하였다. 동일 웹 서버에서 서비스 되는 웹 파일들은 대부분 비슷한 평활상수에서 예측 성능이 최대가 됨을 발견하고 FPLC 전략에서는 한 개의 평활상수만을 이용하여 각각의 파일의 선호도를 예측하는데 사용하였다. 지수평활법이 매우 간단한 알고리즘으로 적용이 가능하기 때문에 FPLC 전략의 시간 복잡도가 PLC 전략과 거의 비슷하면서도, 과거의 요청된 선호도를 사용하는 PLC 전략에 비하여 우수한 성능을 보였다. 또한 선호도의 예측 정확도가 높아질수록 캐시 성능도 민감하게 반응하며 증가함을 확인하였다.

본 연구에서 사용한 지수평활법이 웹 파일의 선호도 예측에 가장 적합한 방법이라고 할 수는 없다. 웹 파일의 선호도에 대한 예측 성능을 높이기 위한 연구를 통해 캐시 적중률의 증가를 기대할 수 있다.

참 고 문 헌

[1] D. W. Chang, H. R. KE, R. C. Chang, "Adaptive-level memory caches on World Wide Web servers," Elsevier Computer.

[2] E. Levy-Abegnoli, A Iyengar, J. Song and D. Dias, "Design and performance of Web server acceleration," Proceedings of Infocom '99, 1999.

[3] E. P. Markatos, "Main memory caching of web documents," Proceeding of the Fifth International World Wide Web Conference, May, 1996.

[4] J. Challenger, A. Iyengar and P. Dantzig, "A scalable system for consistently caching dynamic Web data," Proceeding of Infocom '99, 1999.

[5] Mike Reddy & Graham P. Fletcher, "Intelligent web caching using document life histories : A comparison with existing cache management techniques," J228, School of Computing University of Glamorgan, Pontypridd, Mid Glamorgan. CF 37 1DL.

[6] Stanistaw Walukiewiez, "Integer Programming," Polish Scientific Publishers Warszawa, 1991.

[7] Squid Internet Object Cache, http://squid.nlar.net/.

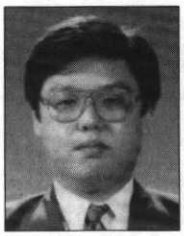
[8] V. N. Padmanabhan, J. C. Mogul, "Using Predictive Prefetching to improve world wide web latency," Computer Communication Review 26, 1996.

[9] 유해영, "웹 서버에서의 선호도를 기반으로 하는 게으른 캐싱 기법", 아주대학교 박사학위 논문, 2001.

[10] 유해영, 박 철, "선호도 기반 웹 캐싱 전략", 정보과학회논문지 : 시스템 및 이론, Vol.29, No.9, 2002.

[11] 최기헌, 이종협, "시계열 분석과 그 응용", 자유아카데미, 1994.

[12] 한국인터넷정보센타, http://stat.nic.or.kr/sdata.html, 2002.



박 철

e-mail : pcman@dankook.ac.kr
1994년 단국대학교 전산통계학과(이학사)
1998년 단국대학교 대학원 전산통계학과 (이학석사)
2001년 단국대학교 대학원 전산통계학과 박사과정 수료
2001년~현재 단국대학교 정보컴퓨터학부 강의전임강사
관심분야 : 웹 어플리케이션, 소프트웨어 공학, 분산 시스템 등

유 해 영

e-mail : yoohy@dankook.ac.kr
1979년 단국대학교 수학과(이학사)
1982년 단국대학교 대학원 수학과(이학석사)
1994년 아주대학교 대학원 컴퓨터공학과 (공학박사)
1983년~현재 단국대학교 정보컴퓨터학부 교수
관심분야 : 소프트웨어 공학, 시스템 프로그램 등