웹 객체 참조 확률분포특성을 이용한 캐싱 기법

나윤지*
*호남대학교 인터넷소프트웨어학과
e-mail:yjna@honam.ac.kr

An Caching Method using the Probability Distribution Characteristics of the Web Object Reference

Yun-Ji Na*
*Dept of Internet Software, Honam University

요 약

대부분의 웹 캐싱 관련 연구들은 객체적중률의 향상과 캐싱 비용의 절감을 중심으로 이루어졌다. 하지만, 웹 객체 참조의 확률분포특성은 웹 캐싱 기법들의 성능을 감소시키는 중대한 원인이 되고 있다. 따라서 웹 캐싱의 성능을 향상시키기 위해서는 객체 참조의 확률 분포특성을 기반으로 한 캐싱 능력 향상에 대한 연구가 필요하다. 본 연구에서는 객체 참조의 확률분포특성 기반의 적응성을 가진 새로운 웹 캐싱 기법을 제안하였다. 또한 실험을 통해 제안기법의 성능 향상을 확인하였다.

1. 서론

웹 캐시의 성능은 한정된 웹 캐시 저장 영역의 효과적인 관리에 달려있다. 이를 위해, 자주 사용되는 웹 객체를 캐시의 저장 영역에 유지하기 위한 대체 기법에 대한 연구들이 활발히 진행되고 있다[1,2,3]. 웹 캐시를 위한 대체 기법은 웹 객체의 특성을 반영하여야 한다. 웹 객체의 사용자 참조특성은 다음과 같이 정리할 수 있다[1].

- 웹 객체는 시간과 지역에 따른 참조 국지성을 가진다. 이러한 참조 특성은 시간에 흐름에 따라 가변적이며 이는 기존 캐싱 기법의 성능을 떨어뜨리는 중요한 요인이 되고 있다.
- 사용자 연령 및 인터넷 사용의 숙달 정도, 교육 수준 등과 같은 사용자 특성은 참조 특성에 영향을 미친다.
- 웹서비스의 형태와 특성은 사용자의 참조 특성에 영향을 미친다.
- 참조 특성의 가변성은 객체 적중률의 편차를 크게 한다.
- 참조특성은 가변성은 비주기적으로 일어난다.

이와 같은 웹 객체에 대한 가변적인 참조특성 변화는 웹 캐싱의 성능을 감소시키는 중대한 요인이다. 하지만 현재까지 웹 캐싱기법들은 이러한 특성을 반영하지 못하고 있어 이에 대한 연구가 필요하다.

본 연구에서는 객체 참조의 확률분포특성 기반의 새로운 웹 캐싱 기법을 제안하였다. 제안기법은 객체 참조의 확률분포특성을 기반으로 객체를 4개의 그룹으로 나누고, 객체 참조의 확률분포특성 변화에 따라 적응적으로 wear out되는 2개의 캐싱영역에 대한 캐싱을 통해 캐싱 능력을 향상시켰다. 실험결과 제안기법이 기존의 기법들에 비해 객체적중률의 향상을 확인할 수 있었다.

2. 제안기법

2.1 시간 흐름에 따른 캐시 적중률의 변화

그림 1은 객체적중률의 변화 특성에 중점을 두기 위해 시간의 흐름에 따른 캐시 적중률의 변화를 데이터 평활화 기법(data smoothing technique)을 사용하여 분석한 그래프이다. 실제 객체 적중률은 그림과 같이 평활화 된 상태가 아니라, 기복과 이상치(out lier) 등이 나타난다. 이러한 데이터를 그래프 형태로 매끄럽게 나타내기 위해서는 전처리를 통해 평활화 하는 과정이 필요하다[16]. 본 연구에서 사용한 평활화 기법은 구간(bin)의 평균값으로 대체하는

구간 평균에 의한 평활화(smoothing by bin means) 기법을 사용하였다. 평활화는 데이터 정제의 한 형태로서 원래의 집합에서 비유사치를 제거하여 데이터를 매끄럽게 하는 기법이다.

그림 1의 그래프에서 t1시점과 t3시점에 적중률의 급격한 하락이 발생하여 평균값 이하로 떨어지며, t2시점과 t4시점에서 적중률이 상승하여 평균 적중률을 유지하게 된다. 이러한 현상이 발생하는 원인은 주로 다음과 같은 사용자의 서핑 형태 변화 특성에 따른 것이다.

▷ 사용자 선호도의 변화
▷ 사용자 웹 서핑 형태의 변화
▷ 사용자의 변화 : 기존사용자 사용 종료, 새로운 사용자 사용 개시

그래프에서 t1 - t2에서의 Δt1과 t3 - t4에서의 Δt2의 넓이의 감소를 통해 웹 캐싱의 성능을 향상시킬 수 있다. 결국 Δt1→ Δt1′, Δt2→ Δt2′ 의 감소& Δw1 → Δw1′, Δw2 → Δw2′의 감소가 웹 캐싱의 성능을 향상 시킬 수 있는 것이다.

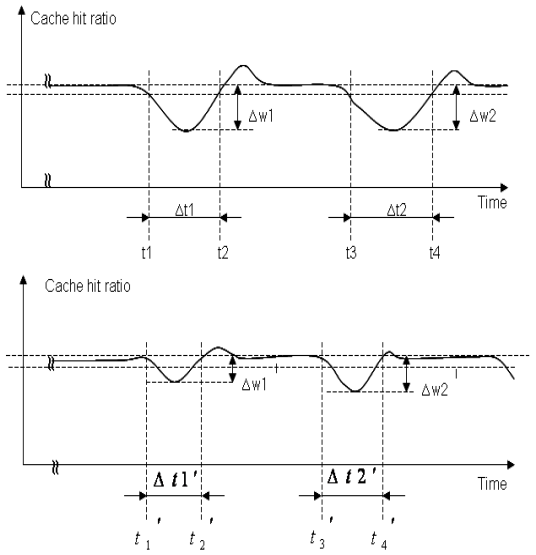

그림 1 참조특성 분석 그래프

2.2 Cache Wear Out(CWO)

정의 1은 OWS(Object Working Set)의 정의를 나타낸 것이다. OWS(i)는 캐시의 시간구간(i)에서 임계값(Threshold Value) 이상의 객체적중률을 유지하기 위해 필요한 웹 객체의 집합이다.

<정의 1> OWS (i)
OWS (i) = { O_1, O_2, …, O_n }, 여기에서 O_1, O_2, …, O_n은 시간구간 (i)에서 임계 값(Threshold Value) 이상의 객체적중률을 유지하기 위해 필요한 객체들

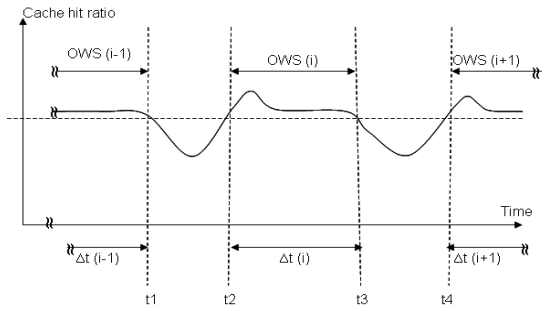

그림 2 Object Working Set

그림 2는 시간 구간(i-1), (i), (i+1) 구간에서의 OWS을 나타낸 것이다. 캐시 적중률이 OWS를 구성하기 위한 임계 값(Threshold value) 이하로 떨어질 경우 CWO(Cache Wear Out)이 발생한다. 그림 3에서 t1과 t3 시점이 CWO 시점(CWO point)이다. 실제 환경에서는 각 시간 구간 내에서도 캐시의 적중률이 자주 임계 값 이하로 떨어지게 된다. 따라서 CWO 시점을 설정하는 것은 임계 값 이하로 떨어지는 횟수를 기반으로 CWO 시점 설정하는 방법과, 각 시간 구간에서의 평균 객체 적중률을 기반으로 일정한 적중률 이하로 떨어지는 시점을 CWO로 설정하는 방법을 사용할 수 있다. 또한 본 논문에서의 접근법은 CWO 시점을 객체참조의 확률 분포함수를 이용하여 결정한다.

CWO가 발생하면 해당 캐시는 캐시를 비우고 새로운 객체를 저장하여 객체적중률을 높이기 위해 새로운 객체 참조 특성을 반영한 OWS를 구성할 때까지 새로 들어오는 객체로 캐시를 채운다. 이 때 CWO로 인해 갑작스런 객체적중률의 하락이 발생한다. 이를 보완하기 위해 다음 절에서 다루는 것과 같이 2중 구조의 캐시를 사용한다. 제안기법에서는 객체참조분포특성을 기반으로 CWO가 발생하며, 이 때 새로운 OWS 구성으로 객체 참조특성 변화에 대한 적응성을 갖고 있다. 이를 통해 캐시 관리에 소요되는 시간을 감소시키며, 객체 대체(Object Replacement) 시간과 객체 검색 시간, 판별 시간을 감소시킨다. 또한 캐시 적중률 향상에 따른 지연 시간 감소와 비용 감소를 가져올 수 있다.

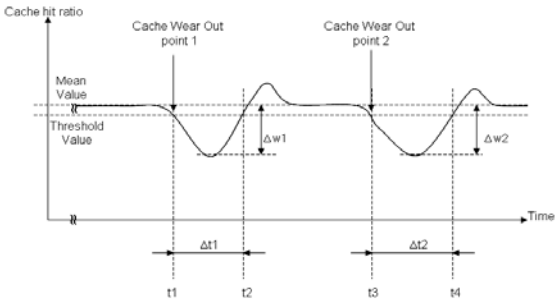

그림 3 Cache Wear Out

2.3 캐시 구조

캐시를 완전히 비웠다가 다시 새로운 객체로 채울 때까지 캐시 적중률의 급격한 하락은 그림 6과 같이 이중 구조의 캐시를 사용함으로서 보완할 수 있다. 객체 적중률이 안정된 시간 구간에서는 두 개의 캐시를 하나의 공간처럼 사용하다가 CWO이 발생하면 한 쪽 캐시 영역을 완전히 비운다. 그리고 비운 영역에 새로 들어오는 객체를 채운다. 이제 비웠던 영역이 새로운 객체로 차고 캐시 부재(cache miss)가 발생하기 시작하면 비우지 않았던 캐시 영역과 하나의 영역처럼 캐시 대체 기법을 적용하여 운영한다. 일정한 시간이 지난 후 다시 CWO가 발생하면 이번에는 이전에 비우지 않았던 한 쪽 영역을 비우고 안정화를 위한 캐시 운영을 계속한다. 그림 4는 이중의 캐시 구조를 위한 이중 캐시 관리자 구조를 나타낸 것이다.

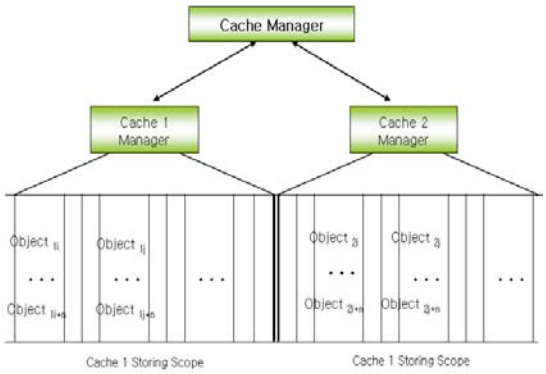

그림 4 캐시 Manager 구조

3. 실험과 분석

3.1 객체 평균 수명

웹 객체는 캐시 내에서 평균 수명을 갖게 된다. 그림 5는 웹 객체의 참조분포 특성에 따라 객체의 크기를 기반으로 4 구간의 아이템으로 분류한 것이다. 아이템1의 구간 S1은 5K 미만, 아이템2의 구간 S2는 5K 이상 100K 미만, 아이템3의 구간 S3는 100K 이상 400K 미만, 아이템4의 구간 S4는 400K 이상 객체의 구간이다.

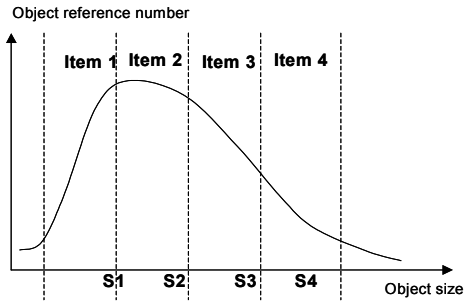

그림 5 객체 Item 분류

4개의 아이템은 S1, S2, S3, S4는 동시에 테스트된다. 이때 실험에서 각 구간에서 발생하는 캐시 부재의 횟수는 다음과 같다.

S1 = 75,123
S2 = 315,634
S3 = 67,473
S4 = 13,935

이것은 평균 객체 적중률을 40%로 하였을 때, 90%인 36% 이하로 객체 적중률이 떨어지는 시점까지의 객체 구간별로 발생한 객체 부재의 횟수이다. 평균 수명 파라미터(θ)는 다음과 같다[4].

$$\hat{\theta} = \frac{T}{r}$$

T = total test time accumulated on all items

r = total number of cache miss

평균 수명은 다음과 같은 카이스퀘어(chi-square, χ^2) 분포 값을 통해 구할 수 있다.

$$P\left[\chi^2_{1-\alpha/2,2r} \le \frac{2r\hat{\theta}}{\theta} \le \chi^2_{\alpha/2,2r}\right] = 1-\alpha$$

이것에 대한 평균 수명은 다음과 같다.

Mean life = 472,165/4 = 118,041

평균 수명에 대해 95% 양측 신뢰구간(two-sided confidence interval)을 구해보면 적절한 카이스퀘어 테스트 값은 다음과 같다.

$$\hat{\theta} = \frac{\sum_{i=1}^{4}\chi_i}{4} = \frac{472,165}{4} = 118,041$$

$$\chi^2_{0.975,8} = 2.180$$

$$\chi^2_{0.025,8} = 17.535$$

$13,464 \leq \theta \leq 108,295$

따라서 이 카이스퀘어 분포 특성을 갖는 테스트에 대해 객체의 평균 수명은 최소 값 13,464회, 최대 값 108,295회를 갖는다. 일반적으로 이 방법은 제품의 평균 수명을 구할 때 사용한다[4]. 즉, 이 최대 값과 최소 값 구간은 제품의 평균 수명 구간이라 할 수 있고, 이 구간에서 제품은 fault state에 도달하게 된다. 제품이 fault state가 되면 교체를 하든가, 갱신(renewal)을 할 수 있다. 이것을 캐시의 관점에서 보면 CWO를 통해 갱신하여 성능을 높일 수 있게 된다.

3.2 에이징을 통한 CWO 정책

본 논문에서는 언제 캐시를 wear out 시키느냐는 문제에 대해 χ^2객체참조분포특성 이용한다. CWO는 객체의 평균 수명을 기준으로 최소 값≤θ≤최대 값 (θ는 평균 수명 파라미터)의 범위 내에서 에이징(aging) 기법을 통해 적응적으로 발생한다. CWO 정책은 다음과 같다.

■ policy1: 최대 값을 기준으로 wear out

객체의 사용이 아주 많은 경우 빈번한 객체 부재가 발생하게 된다. 이 경우 객체의 아이템 구간마다 fault state에 도달하는 시간이 짧아진다. 캐시에서 객체의 fault state에 도달하는 시간이 짧을 때 경우, 하나의 CWO 정책을 사용하게 되면 너무 빈번한 CWO가 발생한다. 따라서 이 경우 최대 값을 기준으로 CWO 시점을 결정한다.

■ policy2: 최소 값을 기준으로 wear out

- 객체의 사용 빈도가 낮은 경우 캐시 부재는 아주 더디게 발생한다. 이 경우에는 평균 수명 구간에서 최소 값을 기준으로 CWO을 발생시킨다.

3.3 Experiment Result and Analysis

그림은 ①기존의 기법의 시간에 따른 크기적중률 변화의 실험 결과와, ②제안기법의 시간에 따른 크기적중률 변화의 실험 결과의 비교를 하나의 그림으로 나타낸 것이다. 그림에서 상단은 기존의 단일 영역 기법의 시간에 따른 크기적중률 변화의 실험 결과이고, 하단은 제안기법에 대한 실험결과를 나타낸 것이다.

두 실험을 비교해보면 객체크기적중률의 급격한 변화의 넓이인 $\Delta t1 \rightarrow \Delta t1'$, $\Delta t2 \rightarrow \Delta t2'$ 의 감소를 확인할 수 있었다. 하지만 제안 시스템의 성능을 더욱 향상시키기 위해서는 넓이만이 아니라 변화의 폭 또한 줄일 수 있어야한다. 즉, 객체크기적중률의 급격한 변화에 대해 변화의 넓이만이 아니라 변화의 폭 감소는 제안시스템의 성능을 더욱 향상시킬 수 있어, 이에 대한 분석과 향후 계속적인 연구가 필요하다.

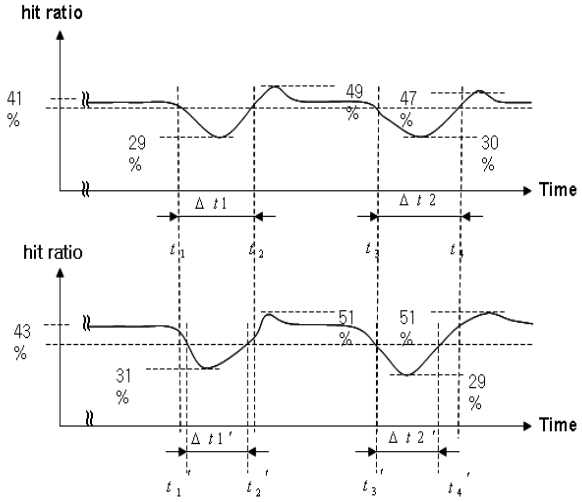

그림 9 실험결과

4. 결론

기존의 웹 캐싱 관련 연구들은 캐싱 기법의 연구를 통해 객체적중률의 향상과 이를 기반으로 캐싱 비용 절감에 대한 연구가 이루어졌다. 본 연구에서는 객체 참조의 확률분포특성 기반의 적응성을 가진 새로운 웹 캐싱 기법을 통해 캐싱 시스템의 성능을 향상시켰다.

참고문헌

[1] Il Seok Ko, Yun Ji Na, Choon Seong Leem, "ACASH: An Adaptive Web Caching Methid with Heterogenity od Web Object and Reference Characteristics," Journal of KISS: Information networking, vol.31, no.3, pp.305-313, 2004.

[2] N. Niclausse, Z. Liu, P. Nain, "A New and Efficient Caching Policy for the World Wide Web," Proc. Workshop on Internet Server Performance(WISP 98), pp.94-107, 1998.

[3] C. Aggarwal, J. Wolf and P. Yu, "Caching on the World Wide Web," IEEE Trans. Knowledge and Data Engineering, vol11, no.1, pp.94-107, 1999.

[4] K. C. Kapur, L. R. Lamberson, *Reliability in Engineering Design*, John Wiley &Sons, 1977.