WWW을 위한 분할 캐쉬에서의 최적 교체 알고리즘

박성주[0] 이은희 이동만
한국정보통신대학원대학교공학부
{lazz, ehlee, dlee}@icu.ac.kr

Optimal replacement policies in partitioned caches for the WWW

Sungju Park[0] Eunhee Lee Dongman Lee
School of Engineering, Information and Communications University

요 약

웹 캐쉬는 자주 쓰이는 웹 문서를 복제함으로써 네트웍 혼잡, 서버 부하, 문서 도착 시간을 줄이는데 목적을 둔다. 웹 캐쉬에서 중요한 이슈 중에 하나인 제한된 저장 공간을 어떻게 사용할 것인가에 대한 연구로서 분할 캐쉬 접근 방법이 있다. 분할 캐쉬는 캐쉬 저장 공간을 여러 개의 분할된 영역으로 나눔으로써, 이질적인 웹상의 객체를 동종의 데이터 집합으로 나누어서 각각의 분할 영역에서 다루도록 할 수 있게 한다. 실험적 연구 결과는 분할 캐쉬가 기존의 캐쉬 저장 공간을 관리하는 교체 알고리즘보다 우수한 성능을 보여준다는 것을 증명하고 있다. 그러나 기존의 분할 캐쉬에서는 각각의 분할 영역에서 동일한 교체 알고리즘을 사용하였다. 본 연구는 각각의 분할 영역에 다양한 교체 알고리즘을 적용하는 실험을 하고, 이 실험 결과에 기반 하여 웹 상에서의 분할 캐쉬를 위한 최적 교체 알고리즘을 제시한다

1. 서론

웹 캐쉬는 빈번한 접속 요구가 있는 웹 문서를 저장시켜 둠으로써 네트웍 혼잡, 서버의 부하, 문서 도착 시간을 줄이는데 그 목적이 있으며 결과적으로 전체적인 WWW의 성능을 향상시킬 수 있다. 이러한 웹 캐쉬는 클라이언트 캐쉬, 서버 캐쉬, 프락시 캐쉬, 계층적 캐쉬, 협력적 캐쉬 [8] 등 여러 접근방법이 있다. 이런 모든 접근방법 들은 보다 나은 hit rate을 얻기 위해 웹 캐쉬의 한정된 저장 공간을 어떻게 효율적으로 관리할 것인지가 공통의 문제이다. 이에 대한 접근 방식의 하나로 새로운 문서를 위해 저장된 문서를 교체하는 교체 알고리즘에 대한 연구가 있으며, FIFO, LRU, LFU, LFF, GDS 등의 많은 교체 알고리즘의 연구가 진행되어져 왔다[3, 6]. 또 다른 접근 방식으로는 웹 환경에서 인식되는 높은 변동성을 고려하면서 캐쉬 공간 관리의 효율성을 더욱 높이기 위한 연구로서 분할 캐쉬(partitioned cache) [5, 6] 방식이 있다.

기존의 분할 캐쉬에 대한 실험결과는 캐쉬 저장 공간을 관리하기 위한 분할 캐쉬 방식이 웹 문서의 다양성을 고려하기 때문에 캐쉬 공간을 분할하지 않는 교체 알고리즘보다 낫다는 것을 보여주고 있다. 분할 캐쉬는 캐쉬 저장 공간을 여러 개의 분할된 영역으로 나눔으로써, 이질적인 웹상의 객체를 동종의 데이터 집합으로 나누어서 각각의 분할 영역에서 다루도록 할 수 있게 한다. 이는 각각 같은 특성을 갖고 있는 데이터 집합의 참조 지역성을 분할 영역 안에서 보존하기 위함이다.기존의 연구[5]에서는 대표적인 교체알고리즘인 LRU와 LFF를 분할 캐쉬의 성능과 비교 분석하여 그 우수성을 증명하였다. 그러나 이 연구에서는 캐쉬의 분할된 각 영역에 같은 교체 알고리즘을 적용함으로써 분할 캐쉬에서 사용 가능한 장점을 다 활용하지 못하였다. 이에 본 연구에서는 각 분할된 영역에 서로 다른 교체 알고리즘을 각각 적용하고, 실험에 의하여 각각의 분할영역에 맞는 최적의 교체 알고리즘을 찾아내어 분할 캐쉬를 위한 최적 교체 알고리즘을 제안한다. 이 알고리즘으로 인하여 WWW의 다양성과 캐쉬 저장공간을 분할함으로써 얻어질 수 있는 장점들을 획득할 수 있으므로 전체적인 WWW을 위한 캐쉬 성능을 더욱 높일 수가 있다.

이 논문의 2장에서는 분할 캐쉬 관련 연구에 관해서 논의할 것이고, 3장에서는 분할 캐쉬 설계를 위한 환경을 설명할 것이며 본 연구에서의 실험적 결과와 분석은 4장에서 논의할 것이다. 끝으로 5장에서는 결론과 앞으로의 연구 방향에 대해서 논의할 것이다.

2. 관련연구

2.1 Web cache를 위한 교체 알고리즘

전통적인 교체 알고리즘들은 일반적으로 참조되는 스트림의 기본 속성 한가지를 의존하여 사용되어져 왔다. LRU(Least-Recently-Used)는 최근에 참조되었던 문서들은 다시 참조될 가능성이 높다는 시간적 지역성에 의존하고 LFU(Least-Frequently-Used)는 이전에 빈번하게 접속된 문서들은 앞으로도 다시 요청 될 가능성이 크다는 문서의 편중된 요청 패턴에 의존을 한다. 반면 LFF(Largest-File-First)는 크기가 작

은 문서가 다시 참조될 가능성이 높다는 문서의 크기와 참조 경향 사이의 상호 역관계에 의존한다. 한편, 전통적인 교체 알고리즘들에 비하여 최근의 여러 연구에서는 문서의 크기와 접근 비용의 다양성과 변동성을 함께 고려하는 여러 LRU 변형 알고리즘이 제시되었다. GreedyDual Algorithm [2] 은 다양한 접근비용과 동일한 크기를 갖는 웹 문서들에 대한 캐쉬 방법으로서, 이는 웹 문서의 다양한 크기를 고려할 수 있게 GDS(GreedyDual-Size) [3] 에서 일반화 되어 제시되었다.

2.2 WWW을 위한 분할 캐쉬

WWW에서 인식된 문서 크기의 높은 다양성을 고려하기 위해 [5, 6]에서 캐쉬 저장 공간을 관리하기 위한 새로운 방법인 분할 캐쉬가 제안되었다. 분할 캐쉬란 모든 문서를 저장하는 단일 캐쉬를 가지는 것 대신, 캐쉬 저장 공간을 여러 개의 분할 영역으로 나누어서 문서들의 크기나 타입에 따라 각각의 문서들을 저장하는 방법이다. 각각의 분할 영역은 문서 크기나 타입에 의해서 분류되어지므로, 분할된 캐쉬 공간은 같은 종류의 문서들을 관리 할 수 있게 되며 그에 맞는 교체 알고리즘의 적용이 가능해진다. 캐쉬를 분할하는 기본적인 이유는 각각의 분할 영역에서 참조 지역성을 보존함으로써 전체 캐쉬 공간의 hit rate을 높이고자 하는 데 있다. 특정 분할 영역에 저장된 문서는 오직 그 문서와 같은 종류의 문서에 의해서만 교체되어지므로, 크기가 작은 문서의 참조 지역성이 크기가 큰 문서 때문에 방해되어 지지 않게 된다. 이는 일반적으로 웹 상에서는 크기가 작은 문서가 상대적으로 높은 요청 빈도를 가지고 있다는 사실에 근거하고 있다.

WWW를 위한 분할 캐쉬 연구는 [6]에서 처음 제안되었다. 그러나 이 연구에서는 단순히 audio와 non-audio의 2가지 타입의 문서를 대상으로 연구되었다. Murta 등은 [5]에서 기존의 교체 알고리즘과의 비교를 통하여 분할 캐쉬의 성능을 분석하고, 수치적 결과에 근거하여 캐쉬 성능을 평가하는데 쓰이는 두 가지 평가 기준 □ hit rate와 byte hit rate □ 모두에서 분할 캐쉬가 더 좋은 성능을 보여준다는 것을 지적하였다. 그러나 이 연구에서는 모든 분할 영역에 동일한 교체 알고리즘을 사용하였기 때문에, 분할 캐쉬의 최적의 성능을 보여주지는 못하였다.

3. 분할 캐쉬 설계를 위한 환경

본 연구에 사용한 실험에서는, 최적 교체 알고리즘을 찾기 위해 모든 분할 영역에 같은 교체 알고리즘을 적용한 후 그 결과를 관찰하였다. 즉, 매번 동일한 trace를 사용하고 매번 다른 교체 알고리즘을 분할 캐쉬에 적용하여, 특정 분할 영역에서 좋은 성능을 보이는 교체 알고리즘들을 종합하여 분할 캐쉬에서의 최적의 교체 알고리즘을 찾아내고자 하였다..

이를 위해 실제 웹 프락시 서버의 로그 파일을 사용하여 여러 교체 알고리즘을 시뮬레이션 된 분할 캐쉬에 적용시켰다.

3.1 Workload 특징

시뮬레이션을 위해 Virginia Tech. 에서 1995년 2월에서부터 4월까지 모아진 로그를 사용했다. 표 1에서 사용한 workload에 대한 통계를 나타냈다. 이를 보면 웹 workload의 높은 다양성과, 크기가 작은 문서에 대한 요청이 일반적으로 더 많은 반면, 전체 전송량은 크기가 큰 문서에 의해서 영향을 받는다는 것을 알 수 있다.

3.2 분할 캐쉬의 설계 요구사항

WWW을 위한 분할 캐쉬를 설계하는데 필요한 고려 사항들은 다음과 같다. 캐쉬를 몇 개로 분할할 것인가, 각 분할 영역의 크기, 각 분할 영역이 저장하는 문서의 크기의 제한범위 등이 그것이다. 이전의 연구에서 실험을 통해 구한 가장 적합한 분할 영역의 크기는 각각 전체 캐쉬 크기의 1/10, 2/10, 7/10을 소(小), 중(中), 대(大) 크기의 문서를 저장하기 위해 할당하는 것이다 [5]. 또한 각 분할 영역에 들어가는 문서의 크기는 [7] 에 있는 웹상의 workload 특징에 기반 하여 각각 (2KB 미만, 2KB ~ 6KB, 6KB 이상) 으로 설정하였다.

크기(Byte)	요청횟수	요청횟수 (%)	요청량 (Mbytes)	요청량 (%)
0	1551	16.61%	0.000	0%
1B□ 10B	0	0%	0.000	0%
10B □ 100B	207	2.22%	0.014	0.01%
100B □ 1KB	1982	21.23%	1.046	0.88%
1KB □ 10KB	4364	46.75%	13.635	11.49%
10KB □ 100KB	1082	11.59%	28.067	23.66%
100KB □ 1MB	133	1.42%	28.805	24.27%
1MB □ 10MB	17	0.18%	47.104	39.69%
총합	9336	100%	118.671	100%

표 1. workload 통계

4. 실험 결과와 분석

이 장에서는 실험 결과를 요약하고, 웹 캐쉬에 주로 쓰이는 두 가지 성능 평가의 기준 □ HR (Hit Rate)와 BHR (Byte Hit Rate) □ 에 근거하여 결과에 대한 분석을 수행하고자 한다.

실험에 쓰인 분할 캐쉬 설계에 관련된 여러 설정들이 표 2에 요약되어져 있다.

분할 방식	분할된 캐쉬 영역의 수 : 3
캐쉬 크기	workload 전체량의 5%
분할 영역의 크기	분할 영역 小 : 전체 크기의 1 / 10
	분할 영역 中 : 전체 크기의 3 / 10
	분할 영역 大 : 전체 크기의 7 / 10
분할 영역의 분류	분할 영역 小 : 문서 크기 < 2 KB
	분할 영역 中 : 2KB < 문서 크기 < 6 KB
	분할 영역 大 : 문서 크기 > 6 KB
교체 알고리즘	FIFO, LRU, LFF, LFU 가 매번 한 번씩 전 분할 영역에 쓰인다

표 2 실험을 위한 설정

설정 중에 하나인 캐쉬의 크기는 시뮬레이션에 사용된 trace 총량의 5%로 고정되어 있다. 캐쉬의 크기가 시뮬레이션에서 쓰인 trace 전체 총량과 같을 때, 이러한 캐쉬를 ' 무한 캐쉬' 라고 볼 수 있고, 이 캐쉬에서는 어떤 문서도 교체되지 않을 것이다. 무한 캐쉬에서는 모든 교체 알고리즘이 같은 성능을 보일 것이고, 따라서 같은 hit rate을 보이게 될 것이다. 캐쉬 크기가 작을 때 더 좋은 성능을 보일 수 있는 교체 알고리즘이 더 우수한 알고리즘임은 명백하기 때문에, 각각의 교체 알고리즘의 성능 차이를 충분히 나타낼 수 있도록 이 실험에서는 캐쉬의 크기를 workload 총량의 5%로 설정했다.

4.1 성능 평가 기준

캐쉬에 쓰이는 교체 알고리즘을 평가할 때에는 HR와 BHR 중 어느 쪽을 기준으로 하느냐에 따라 확연히 다른 결과가 나온다[6]. 이는 각각의 교체 알고리즘은 각각 다른 환경에서 더 뛰어난 성능을 발휘하기 때문이므로, 이 실험에서는 두 가지 측정 기준을 모두 고려하여 분할 캐쉬에 적합한 교체 알고리즘을 선택하기로 한다.

4.2 결과 및 분석

표 3과 4는 시뮬레이션 결과를 보여준다. 표 3은 3개의 분할 영역에 각각의 교체 알고리즘이 적용되었을 때의 각 분할 영역에서의 교체 알고리즘의 HR과 그를 종합했을 때의 전체 캐쉬의 HR을 보여주고 표 4에서는 BHR을 보여주고 있다. 표에 의하면 분할영역 小와 中에서는 LRU가 가장 좋은 성능을 보여주고 있고, 분할영역 大에서는 LFF가 가장 좋은 성능을 보여주고 있다.

	전체캐쉬	분할영역小	분할영역中	분할영역大
FIFO	31.53%	42.34%	27.79%	12.81%
LRU				13.47%
LFF	33.08%	44.16%	25.74%	
LFU	28.50%	39.56%	21.32%	13.29%

표 3. Hit Rate

	전체캐쉬	분할영역小	분할영역中	분할영역大
FIFO	10.54%	41.59%	26.90%	8.65%
LRU	11.68%			9.69%
LFF		40.52%	23.28%	
LFU	9.89%	39.12%	20.45%	8.40%

표 4. Byte Hit Rate

이 실험결과를 각 알고리즘의 특성과 분할 캐쉬의 원리에 의해서 다음과 같이 분석하였다.

첫째, 분할 영역 小와 분할 영역 中을 위한 가장 좋은 교체 알고리즘은 LRU이다. 일반적으로 웹 캐쉬에서는 LFF가 LRU보다 더 나은 성능을 가진다고 알려져 있다[6]. 그러나 위의 실험결과는 분할 캐쉬의 小, 中 분할영역에서는 LRU가 LFF보다 더 뛰어난 성능을 가지고 있다는 것을 보여준다. 이는 분할 캐쉬가 이미 LFF의 특징을 충분히 활용하고 있다는 사실에 기인한 것이다. LFF의 기본 원리는 웹 상에서 비교적 낮은 요청 빈도를 가지는 크기가 큰 문서를 먼저 교체시키는 것이다(Large-File First). 그리고 분할 캐쉬는 높은 빈도로 요청되는 작은 크기의 문서가 큰 크기의 문서로 인하여 교체되지 않도록 문서의 종류(크기)별로 캐쉬 공간을 분할하는 것이다. 따라서 LFF가 분할 캐쉬에 쓰인다면 문서 크기와 요청 빈도간의 상호 역관계를 중복하여 적용시키는 것이 될 것이다. 결론적으로, LRU는 분할 캐쉬에 LRU의 특성인 시간적 참조성을 추가함으로써, 이미 활용되고 있는 분할 캐쉬의 특성인 문서 크기와 참조빈도의 역 관계성에 더해져서 小, 中 분할 영역에서의 최적 교체 알고리즘을 제공한다.

둘째, 분할 영역 大를 위한 가장 좋은 교체 알고리즘은 LFF이다. 이는 앞의 결론과 상반되는 듯이 보인다. 그러나 분할 영역 小와 분할 영역 中이 각각 2KB 이하의 문서와 2KB 이상 6KB 미만의 문서를 저장하는데 반하여, 분할 영역 大는 상대적으로 넓은 영역인 6KB 이상의 모든 문서를 저장하고 있다. 따라서 분할 영역 大는 분할 캐쉬가 아닌 일반 캐쉬의 상황과 비슷하여지고, 여기서는 LFF가 LRU보다 더 좋은 성능을 보이게 된다[6]. 따라서, LFF는 분할 캐쉬의 大 분할 영역에서 최적화된 교체 알고리즘을 제공한다.

이상의 결과와 분석을 종합해 볼 때, 분할 영역 小와 中에서는 LRU를, 분할 영역 大에서는 LFF를 교체 알고리즘으로 쓴다면 WWW에서의 분할 캐쉬는 최적화 되어질 수 있다. 이를 분할 캐쉬에서의 '최적 교체 알고리즘'이라고 명명하고, 그림 1에서 최적 교체 알고리즘과 다른 교체 알고리즘의 성능 분석을 보여주고 있다.

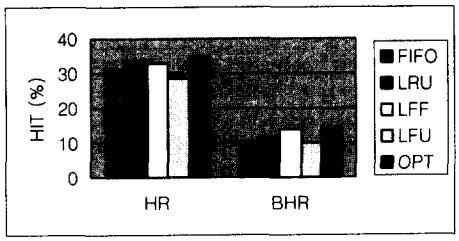

그림 1. 분할 캐쉬에서의 교체 알고리즘의 성능 비교

5. 결론 및 연구방향

본 연구에서는 분할 캐쉬에서 각각의 분할 영역에 적용될 수 있는 교체 알고리즘을 분석하고, 실험결과에 기반 하여 웹 상에서의 분할 캐쉬를 위한 최적 교체 알고리즘을 제시하였다. 두 가지 성능 측정 평가 기준 (hit rate와 byte hit rate) 에 근거하여 제안된 교체 알고리즘은 분할 캐쉬에서 최적의 성능을 보여 줌을 알 수 있다. 향후 연구방향으로는 시뮬레이션에 GDS 알고리즘을 추가하여 최적 알고리즘을 찾아낼 예정이며, 이는 네트웍의 비용을 고려할 수 있는 환경을 필요로 한다.

6. 참고 문헌

[1]. Carlos Cunha, Azer Bestavros, and Mark Crovella. Characteristics of WWW client -based traces. Technical Report BUCS-95-010, April, 1995

[2] Neal E. Young. "*On-line caching as cache size varies*". In Proceedings of Symposium on Discrete Algorithms, January, 1991.

[3] Pei Cao and Sandy Irani. "Cost-aware WWW proxy caching algorithms". In Proceedings of USENIX Symposium on Internet Technology and Systems, December, 1997.

[4] Paul Barford, Azer Bestavros, Adam Bradley, and Mark Crovella. "Changes in Web client access patterns: characteristics and *cacing implications*". *WWW Journal*, 1999.

[5] Cristina Duarte Murta, Virgilio Almeida, and Wagner Meira, Jr. "Analyzing performance of partitioned caches for the WWW". In Proceeding of the 3 rd International WWW Caching Workshop, June, 1998.

[6] Stephen Williams, Marc Abrams, Charles R. Standridge, Ghaleb Abdulla, and Edward A. Fox. "Removal policies in network caches for World -Wid Web documents". *In Proceedings of the ACM* SIGCOMM '96 Conference, Stanford University, CA, August, 1996.

[7] M.F. Arlitt. "A performance study of Internet Web servers". Master's Thesis, Department of Computer Science, University of Saskatchewan, Saskatoon, Canada, 1996.

[8] Jia Wang. "A survey of web caching schemes for the Internet". ACM Computer Communication Review, October, 1999.

[9] Shudong Jin and Azer Bestavros. "*Popularity-aware* greedydual-size algorithms for Web access". In Proceedings of the 20 th International Conference on Distributed Computing Systems(ICDCS), apr, 2000.

[10] Jaeyeon Jung, Dongman Lee and Kilnam Chon. "Proactive Web Caching with Cumulative Prefetching for Large Multimedia Data". 9th WWW conference.