웹 서버 작업부하 감소를 위한 캐시 정책

Cache Policies for WWW Servers to Reduce Workload

임 재 현*
Jaehyun Lim

요 약

본 논문에서는 웹 서버 성능에 있어 캐싱 정책의 영향을 분석하고 연구하였다. 새롭게 제안한 파일 타입 기반 캐싱 정책은 캐시안의 크고 작은 파일 간에 균형 잡힌 결과를 갖도록 지원하며, 뿐만 아니라 작은 파일과 큰 파일에 대한 요청에 훌륭한 성능을 나타낸다. 본 논문에서는 파일 타입 기반 캐싱이 적중률과 바이트적중률 모두 다 좋은 결과를 나타냄을 보인다.

Abstract

In this paper we study and analyze the influence of caching strategies on the performance of WWW servers. We propose a new strategy called file type based caching that aims to obtain a well-balanced mixture between large and small files in the cache, and moreover, it provides good performance for both small and large file as expected. By using the type based caching good results are obtained for both the hit rate and the byte hit rate.

⌐ Keyword : caching, web server, hit ratio, byte hit ratio

1. 소 개

웹(Web)이 직면하고 있는 규모의 문제를 비용면에서 해결하는 가장 효율적인 정책은 데이터를 캐시해서 액세스 지연을 개선하고, 망과 서버의 부하를 줄이는 것이다. 최근 웹 서버에서 사용자 액세스 패턴을 보면 수년전부터 연속 미디어인 오디오와 비디오 파일이 웹상에서 급속하게 증가함을 보여주고 있다[3]. 현재까지의 웹 캐싱은 텍스트와 이미지 데이터를 위해 설계되었기 때문에 연속 미디어인 멀티미디어를 캐싱 하거나 사용자들의 다양한 서비스 요구를 효과적으로 처리하기 위해서는 서버에 대한 액세스 패턴을 정확히 분석 할 필요가 있다. 캐시의 교체 정책은 새로운 파일을 저장하고, 기존의 파일을 제거하기 위해 사용되며, 교체 정책 성능은 캐시구조에서 상당히 중요하다. FIFO, LRU, LFU, SIZE 같은 교체 정책이 분석되고 평가되었으며[4], 기본적으로 캐시의 성능은 적중률(hit ratio)과 바이트적중률(byte hit ratio) 2가지로 측정한다. 웹 캐시는 고정된 크기의 페이지 단위를 캐시 하는 것이 아니라, 가변성을 갖는 파일 단위로 캐시 하기 때문에 단순히 적중률만을 가지고 평가 하기는 어렵다. 이를 보완하기 위해 바이트적중률을 이용한다. 본 논문에서 연구한 웹 캐싱은 프락시 서버 캐싱이 아닌 웹 서버 캐싱을 대상으로 한다. 프락시 서버 캐싱은 프락시 서버의 디스크를 이용하여 액세스된 파일을 저장하고 적중된 파일을 서비스하지만, 웹 서버 캐싱에서는 메인 메모리 공간에서 캐싱이 이루어진다는 차이점이 있다[6,7]. 웹 서버의 메인 메모리 캐싱은 프락시 서버 캐싱보다 데이터를 액세스할 때 월등한 성능을 나타낸다. 멀티미디어 파일의 사용이 증가하고 있는 웹 캐싱에서 캐시해야 할 파일의 크기는 수십 바이트에서 수백 메가바이트로 되어 있으며, 파일의 참조 경향도 매우 다르다. 결과적으로 크기가 매우 크면서 사용

* 종신회원 : 공주대학교 멀티미디어정보·영상학부 교수
defacto@kongju.ac.kr(제 1저자)

가능성이 적은 파일을 저장하기 위해 캐시 교체 정책은 앞으로 자주 사용하는 수천 개의 작은 파일을 교체하는 상황을 만들어 낼 수도 있다. 이는 웹 캐싱에서 바이트적중률을 감소시킨다. 본 논문에서는 멀티미디어 웹 서버에서 발생하는 이 같은 문제점을 고려하여 선행 연구들을 기반으로 하여 파일 타입에 따른 캐시를 사용한다. 성능평가는 빈도수 기반(LFU), 최근참조 기반(LRU)과 크기 기반(SIZE)의 대표적 알고리즘을 사용하여 본 논문에서 사용한 파일 타입별 캐싱 정책과 기존의 통합 캐시 정책과 비교 평가한다. 시뮬레이션에서는 멀티미디어 파일의 사용이 많은 웹 서버와 상대적으로 사용이 적은 웹 서버의 실제 로그를 사용한다. 캐시 크기에 따른 파일 타입별 적중률과 바이트적중률을 평가하여 그 효과를 입증하고 파일 타입 캐싱이 멀티미디어 사용이 많은 웹 서버에서 적중률과 바이트적중률을 모두 개선시킴을 보인다.

2. 관련연구

웹 캐싱은 몇 가지 특별한 특성이 있는데 전통적인 캐싱 방법과는 구별되는 흥미로운 연구 영역이다. 일반적으로 캐시 되는 파일의 크기는 매우 크다. 때문에 주기억장치 캐싱처럼 고정된 크기의 페이지는 가정할 수 없다. 부가적으로, 파일은 서로 다른 타입일 수 있기에, 캐싱 결정에 영향을 미친다. 자주 사용되는 특별한 파일에 대해 비용을 사전에 계산하는 것은 어려운 일이다. 이들 비용은 파일마다 다르고, 심지어 같은 파일도 원래 서버의 부하, 그것의 운영 상태, 클라이언트와 서버간의 거리에 의존한다. 또한 캐시안의 파일은 읽기 전용이며 쓰기 절차가 전혀 필요 없다.

파일 캐싱이 가능한 3지역이 있다. 클라이언트 캐싱은 많은 클라이언트의 요청 속에 있는 지역성이 네트워크 트래픽과 응답시간을 감소하기 위해 클라이언트 측의 캐싱에 의해 연구되었다. 프락시 서버 캐싱은 전형적으로 프락시 서버는 LAN과 인터넷 사이에 위치한다. 클라이언트는 모든 HTTP 요청이 프락시에게 가도록 브라우저를 설정한다. 이렇게 하여, 외부 대역폭이 절약이 될 수 있지만, 프락시 서버의 위험은 그것 자체가 병목이 될 수 있다. 웹 서버 캐싱은 주기억장치에 파일을 저장하여, 디스크 입출력을 감소시킨다. 또한 웹 서버의 액세스 패턴은 대중적인 파일에 일시적으로 집중되는 경향이 있기 때문에 캐시의 사용이 매우 효과적이다.

과거 몇 년 동안 많은 잘 알려진 캐싱 정책이 평가되었다. 이들 정책의 목적은 캐시 적중률과 바이트적중률을 개선시키는 것이다. 모든 접근 방법의 중심은 캐시가 이미 가득 차서 새로운 파일을 저장하려고 할 때 어떤 파일을 교체해야하는지가 의문이었다. 앞서 언급된 알고리즘들은 특별한 상황에 대해 개발되었기 때문에, 하나의 상황에 적합한 알고리즘이 다른 상황에도 좋은 결과를 내는데 실패할 수도 있다. 이것은 캐싱 알고리즘이 어떤 파일 특성을 더 중요하게 보느냐에 있다. 일반적으로 크기 기반 정책(SIZE)이 적중률에서 가장 좋은 결과를 나타내며, 반면에 빈도수 기반 정책(LFU)이 바이트적중률을 개선시킨다[1]. 근본적으로 가장 우수하게 인식되는 정책은 없으며, 오히려 좋은 정책의 선택은 작업부하의 특성에 의존한다. 이 같은 고려사항이 파일 타입 기반 캐싱 정책을 개발한 동기이다.

3. 액세스 패턴 분석

웹 작업부하의 분석은 여러 가지 특성을 파악할 수 있다. 사용자가 액세스 하는 파일의 크기와 타입 분포의 특징, 각 파일 타입에 대한 파일 크기 분포의 특징, 그리고 다른 그룹의 사용자도 같은 액세스 패턴을 가지는가를 분석할 수 있다[8]. 본 논문에서는 웹 작업부하 분석을 통해 인터넷 캐시에서 캐시 성능을 좌우하는 요소를 파악하고 효율적인 캐싱 전략을 제안한다.

3.1 작업부하

웹의 통신량 특성을 파악하기 위해 본 연구에서는 2종류의 작업부하를 분석하였다. 표 1에서 보듯이 서버1은 이미지와 텍스트 사용 비율이 높은 웹 서버이고, 서버2는 멀티미디어 파일의 사용이 높은 웹 서버이다. 파일 타입에 따른 사용자 액세스 요청횟수와 전송바이트 비율을 나타낸다.

〈표 1〉 작업부하(백분율)

파일 타입	서버1 요청횟수	서버1 전송바이트	서버2 요청횟수	서버2 전송바이트
image	76.18	26.59	64.29	0.25
text	16	9.81	9.67	0.13
video	0.23	9.54	1.35	91.24
audio	0.36	38.46	0.93	0.01
application	2.28	9.32	3.99	0.49
etc	4.95	6.28	19.77	7.87
전체	100% (1,075,365)	100% (19,601MB)	100% (92,907)	100% (37,526MB)

서버 1은 요청횟수가 100만 건이 넘지만 전송 바이랑은 19GB이고, 서버2는 요청횟수가 1/10에 불과하지만 전송 바이트량은 37GB가 넘는다. 이것은 서버2가 파일 크기가 큰 비디오의 사용이 많기 때문이다. 텍스트 타입에는 "html, htm, txt"와 같은 확장자를 가진 파일을 포함하고, 이미지 타입에는 "gif, jpeg, xbm" 등의 확장자를 가진 파일을 포함한다. 오디오 타입에는 "au wav, mp3" 등을 포함하고 비디오 타입에는 "mpeg, mpg, qt, mov, avi, movie" 등을 포함한다. 어플리케이션에는 "pdf, ps, latex" 등의 확장자를 가진 파일을 포함하고 나머지는 기타(etc)로 분류하였다.

3.2 파일 크기

파일의 최대 크기는 수십 메가이며, 이는 평균치의 수백 배에 이른다. 파일 크기에 대한 액세스 분포는 평균치보다 작은 것에 집중되어 있고, 몇몇의 아주 큰 파일들이 존재한다. 캐시에 크기가 큰 파일을 저장하지 않고 사용자가 많이 사용하는 크기가 작은 파일을 저장한다면 캐시 적중률은 상당히 증가할 것이다. 그러나 크기가 큰 파일이 캐시에서 적중되면 바이트적중률이 많이 증가한다. 파일의 평균 크기, 중간 크기는 모든 서버에서 유사한 경향을 갖는다.

표 1에서 파일 타입별 요청 및 전송 바이트율을 살펴보면, 이미지와 텍스트가 요청 횟수와는 다르게 절대적인 비율을 차지하지 않는다. 왜냐하면 텍스트 파일의 크기는 다른 타입의 크기보다 작기 때문에 요청 횟수가 많더라도 전체 전송 바이트량은 크기가 큰 다른 타입의 파일에 비해 적게 나타난다. 서버1에서 이미지와 텍스트 타입의 요청이 전체 요청횟수의 92%를 차지하지만, 전송 바이트량은 36% 불과하다. 서버2에서 이미지와 텍스트 타입의 요청이 75%를 차지하지만, 전송 바이트량은 0.4%에 불과하다. 오히려 요청횟수의 1%에 불과한 비디오 타입의 전송량이 전체의 91%를 넘고 있다. 이와 같이 웹 서버의 사용 용도에 따라 파일의 요청횟수와 전송 바이트량에 큰 차이가 있기 때문에, 멀티미디어 파일의 사용이 많은 웹 서버에 기존의 캐싱 정책을 그대로 사용한다는 것은 큰 문제가 될 수 있다.

〈표 2〉 타입별 파일의 평균 크기(단위:바이트)

파일 타입	서버1	서버2
image	5,298	1,577
text	10,024	5,484
video	834,080	27,490,940
audio	2,026,625	4,884
application	78,894	49,538
etc	18,321	149,778

표 2에서 타입별 파일의 평균 크기를 보면, 타입에 따라 많은 차이가 있다. 이미지와 텍스트 파일의 평균 크기는 1K - 10K 정도이다. 반면에 비디오 타입의 파일은 800K - 27M, 오디오 타입의 파일은 4K - 2M 의 평균 크기를 보인다. 이처럼 파일 타입은 파일 크기와 밀접한 관계가 있다. 비디오나 오디오의 파일의 크기는 텍스트나 이미지 파일의 크기보다 상당히 크다. 캐시 크기가 제한되어 있을 때, 비디오나 오디오 파일 대신에 텍스트나 이미지 타입의 파일을 저장한다면 보다 많은 파일을 저장할 수 있을 것이다. 또한 하나의 큰 비디오 파일은 수많은 텍스트 파일을 제거할 수도 있다. 만약 캐시에 크기가 큰 몇몇 파일 만 존재한다면 적중률은 상당히 감소할 것이다. 그러나 크기가 큰 파일이 캐시 내에서 적중된다면, 바이트적중률은 상당히 증가한다. 캐시는 적중률과 바이트적중률을 상호 보완하여 파일들을 저장할 필요가 있다.

4. 제안된 캐싱 정책

액세스 패턴 분석을 기반으로 새로운 캐싱 전략을 사용한다. 파일 타입에 따라 파일 크기는 매우 다르고, 웹 캐싱은 멀티미디어 파일을 연속으로 저장하기 때문에, 기존의 통합된 캐시 대신에 파일 타입별로 캐시를 구성한다.

4.1 캐시 최대 성능

먼저, 캐시 크기를 제한하지 않고 실험함으로써 실험 작업부하의 최고 적중률과 최고 바이트적중률을 구하였다. 표 3을 보면, 실험 작업부하의 적중률은 96% - 98%까지 나타나며 바이트적중률은 94% - 98%이다. 웹 서버의 적중률과 바이트적중률이 높은 이유는 파일 자원의 개수가 한정되어 있고, 그것을 다수의 사용자가 집중적으로 사용하기 때문이다.

〈표 3〉 캐시 최고 적중률 및 바이트적중률

	적중률	바이트 적중률
서버1	96.9	94.3
서버2	98.4	98.6

본 논문에서는 캐시에 저장할 필요성이 있는 두 번 이상 참조된 파일의 총 크기를 구하여 캐시의 전체 크기로 결정하였으며, 앞으로 이 크기를 "유효 캐시 크기"라는 용어로 사용한다. 서버1의 경우 유효 캐시 크기는 860MB이며, 서버2의 경우 유효 캐시 크기는 750MB이다.

4.2 캐싱 전략

액세스 패턴 분석을 보면, 인터넷에서는 다양한 타입의 파일들이 존재하고 타입마다 갖고 있는 특성이 다르다. 그러나 지금까지 인터넷 캐시의 전략은 파일을 타입에 따라 구분하지 않고 모든 파일을 동일하게 취급함으로써 비효율성 문제를 발생시켰다. 특히 멀티미디어 파일의 사용이 많은 웹 서버 캐시에서 크기가 큰 비디오 파일이 크기가 작은 수십 또는 수백 개의 텍스트 파일을 교체한다면, 캐시 적중률을 감소한다. 반대로 크기가 작은 텍스트 파일이 크기가 큰 비디오 파일을 교체한다면, 상대적으로 바이트적중률은 감소한다. 본 논문에서는 이 같은 문제점을 해결하기 위해 파일 타입에 따라 분할된 캐싱 전략을 사용하여, 캐시 적중률과 바이트적중률 모두를 향상시킨다.

파일 타입별 캐싱 전략을 사용할 때, 파일 타입에 따른 캐시 크기를 결정하는 것이 중요한 문제이다. 인터넷에는 다양한 타입의 파일이 존재하고, 각 타입은 캐시의 성능에 영향을 미치고 있다. 크기가 작은 텍스트, 이미지 파일 타입은 캐시의 적중률에 유리하게 작용하고, 상대적으로 크기가 큰 오디오, 비디오 파일 타입은 캐시의 바이트적중률에 유리하게 작용한다. 지금까지의 인터넷 캐시 전략은 모든 파일을 동일하게 취급하였지만, 본 논문에서는 파일 타입별로 캐싱을 수행

한다. 파일 타입별로 캐시를 분리할 때 캐시 크기는 표 1의 사용자 요청횟수와 전송바이트량의 산술 평균을 사용하였다. 표 4는 타입별 캐시의 크기 비율이다. 캐시를 총 6개의 부분으로 구분하였으며, 전체 캐시 크기는 유효 캐시 크기의 10%에서 30% 까지 성능을 측정하였다.

〈표 4〉 타입별 캐시의 크기 비율

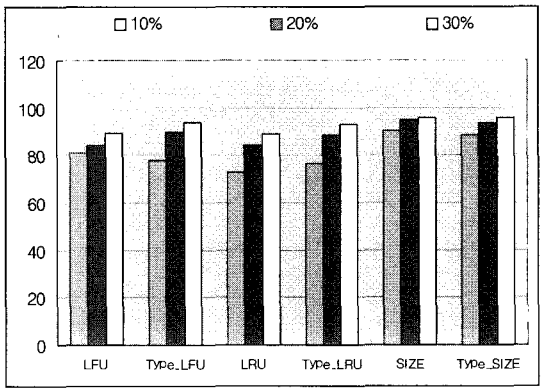
캐시 타입	서버1	서버2
image cache	51.4	32.27
text cache	12.9	4.90
video cache	4.9	46.30
audio cache	19.4	0.50
application cache	5.8	2.20
etc cache	5.6	13.80

5. 평 가

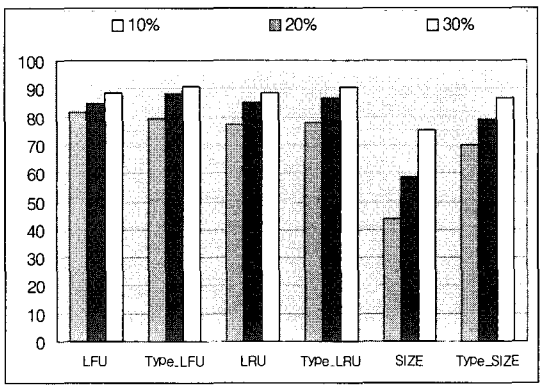
본 논문에서 성능을 평가하기 위해 서로 다른 특성을 가진 2개의 웹 서버에 액세스된 로그 파일을 분석 평가하였다. 시뮬레이션 프로그램은 자바 언어를 사용하여 구현하였고, 시스템은 Dell Powerage 660워크스테이션을 사용하였다. 기존의 캐시 정책(LFU, LRU, SIZE)과 본 연구에서 제안한 파일 타입 기반 정책(Type_LFU, Type_LRU, Type_SIZE)을 비교하였다. 범례에 있는 수치는 유효캐시 크기의 10%에서 30%까지를 의미한다. 선행연구[1]과 마찬가지로 적중률에서는 크기 기반 정책인 SIZE가 가장 우수하고, 바이트적중률에서는 빈도수 기반 정책인 LFU가 우수하다. 반면에 SIZE 정책의 바이트적중률은 매우 떨어진 결과가 나온다.

5.1 서버1 평가

서버 1은 멀티미디어 파일의 사용이 적은 웹 서버이다. 그림 1의 적중률을 보면 기존의 통합 캐시를 사용한 경우 정책에 따라 성능의 차이는 크게 나타나지 않는다. 하지만 파일 타입에 기반한 정책을 LFU와 LRU를 비교해 보면 5% 이상의 성능 개선 효과가 있다. 이것을 수치적으로 분석하면, LFU와 LRU의 유효캐시 크기를 30%로 설정한 것과 Type_LFU와 Type_LRU의 캐시 크기를 20%로 설정한 결과를 비교했을 때 타입 기반 캐싱이 더 우수한 성능을 갖는다.



〈그림 1〉 서버1의 적중률



〈그림 2〉 서버1의 바이트적중률

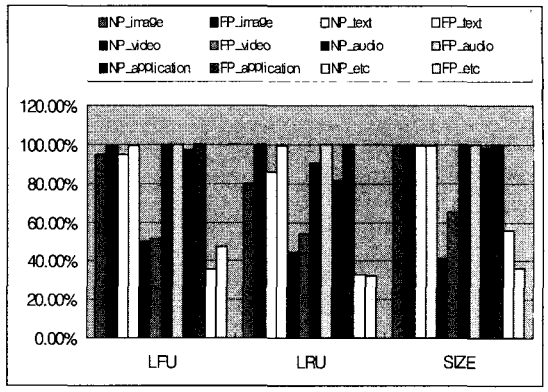
그림 2에서 파일 타입에 기반한 정책을 LFU와 LRU의 바이트적중률과 비교해 보면 3% 이상의 성능 개선 효과가 있다. 특히 SIZE 정책에서는 파일 타입 기반 정책이 설정된 캐시 크기에 따라 10% ~ 20%까지의 성능 개선을 나타낸다. 적중률과 마찬가지로 LFU, LRU와 SIZE의 유효캐시 크

기를 30% 로 설정한 것과 파일 타입 기반 캐시의 크기를 20% 로 설정한 결과를 보면 타입 기반 캐싱이 더 우수한 성능을 갖는다. 결론적으로 메모리 캐시 공간이 1/3 이상 절약되는 효과를 갖으면서 성능은 더 우수하다.

5.2 서버2 평가

서버 2는 멀티미디어 파일의 사용이 많은 웹 서버이다. 그림 3의 적중률을 보면 파일 타입에 기반한 정책을 LFU 정책과 비교하면 6% 이상, LRU 정책과 비교하면 10% ~ 15% 이상의 성능 개선 효과가 있다. 수치적으로 계산하면, LFU와 LRU의 유효캐시 크기를 30%로 설정한 것과 Type_LFU와 Type_LRU의 캐시 크기를 10%로 설정한 결과를 비교 했을 때 타입 기반 캐싱이 더 우수한 성능을 갖는다. 캐시 공간으로 보면 2/3가 절약된 결과이다. 반면에 SIZE 정책에서는 약간 떨어지는 결과를 갖는다. 이것은 크기가 큰 멀티미디어 파일의 사용이 많아서 나타난 현상이다. SIZE 정책은 크기가 큰 파일을 먼저 교체하는 전략이기 때문에 기존의 통합 캐시가 좀 더 많은 작은 파일을 담을 수 있기에 유리하게 작용한 것이다.

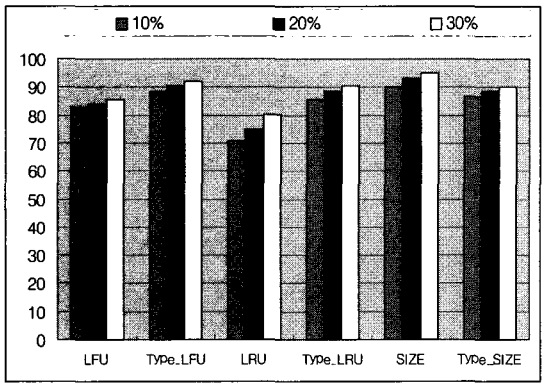
그림 4는 그림 3을 세분화하여 유효캐시 크기 10% 에서 파일별 적중률을 나타낸다. NP는 통합



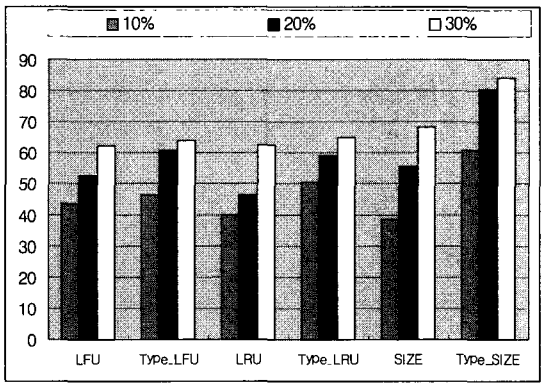
〈그림 4〉 파일별 적중률(유효캐시크기 10%)

캐시의 파일을 의미하며 FP는 파일 타입별 캐시의 각 파일을 의미한다. 기존의 통합 캐시에서는 상대적으로 LRU의 성능이 떨어지지만, 파일 타입별 캐시는 모든 정책에서 성능이 고르게 나타나고, 파일별 성능에서도 우수한 결과를 나타낸다. 파일별로 분석하면 이미지, 텍스트, 오디오, 어플리케이션에서는 100% 에 근접하는 성능을 나타낸다. 비디오 파일인 경우 제안하고 있는 파일 타입별 캐싱 정책이 약간 우수한 결과를 나타내고 있지만, 파일의 크기가 매우 크기 때문에 바이트 적중률에 큰 영향을 미친다.

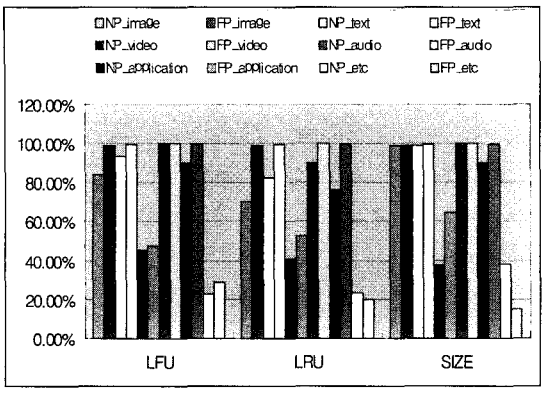
그림 5의 바이트적중률을 보면 파일 타입에 기반 한 정책을 LFU 정책과 비교하면 2% ~ 8% 이상, LRU 정책과 비교하면 3% ~ 10% 이상의 성능



〈그림 3〉 서버2의 적중률



〈그림 5〉 서버2의 바이트적중률



〈그림 6〉 파일별 바이트적중률(유효캐시크기 10%)

개선 효과가 있다. 특히 SIZE 정책과 비교하면 적중률과는 다르게 20% ~ 25% 이상의 성능 개선 효과가 있다. 이것은 멀티미디어 파일의 적중률이 상대적으로 높기 때문에 필연적인 결과이다. 적중률과 마찬가지로, 기존 통합 캐시의 크기를 30%로 설정한 것과 본 논문에서 제안한 방법으로 캐시 크기를 10% 로 설정한 결과를 비교 해보면 타입 기반 캐싱이 더 우수한 성능을 갖는다.

그림 6은 그림 4를 세분화하여 유효캐시 크기 10% 에서 파일별 바이트적중률을 나타낸 것이다. 적중률과 마찬가지로 파일 타입별 캐시는 모든 정책에서 성능이 고르게 나타나고, 파일별 성능에서도 우수한 결과를 나타낸다. 특히 통합 캐시의 SIZE 정책과 비교하면 괄목할 만한 성능의 개선이 나타난다. 특히 비디오 파일인 경우 70% 이상의 성능 개선이 나타나기 때문에 전체 바이트적중률에 큰 영향을 미친다. 파일별로 분석하면 이미지, 텍스트, 오디오, 어플리케이션에서는 100% 에 근접하는 성능을 나타내며, 비디오 파일인 경우 큰 폭의 성능 개선 효과를 나타낸다. 이것이 파일 타입별 정책의 가장 큰 장점 중의 하나이다.

6. 결 론

캐싱의 장점은 참조 지역성과 시간 지역성을 유지하여 적중률과 바이트적중률을 높이는 것이다. 현재 많이 사용하고 있는 대부분의 교체 정책은 빈도수 기반 정책, 최근참조 기반 정책, 크기 기반 정책 중 하나를 모태로 하여 사용하고 있다. 하지만 이들 수많은 알고리즘과 정책들은 적중률과 바이트적중률을 동시에 만족시키지 못하고 있다. 특히 멀티미디어 파일의 사용이 증가하면서 바이트적중률이 점점 떨어지는 양상을 보이고 있다. 일반적으로 적중률이 높다는 것은 작은 파일에 유리하게 작용하는 정책이며, 바이트적중률이 높다는 것은 큰 파일에 유리한 정책이라는 결론이 나온다. 이 양자를 병립하기가 매우 어려운 실정이다. 본 논문에서 사용한 방법은 적중률 또는 바이트적중률에 있어 어느 한 가지 최고의 성능을 갖는 정책 보다는, 양자에 있어 우수한 성능을 나타내는 정책에 초점을 갖고 실험을 하였다. 일반적인 성격의 웹 서버와 멀티미디어 파일의 사용이 높은 웹 서버를 대상으로 실험하였으며, 기존의 캐시와는 다르게 파일 타입별로 캐시를 설정하여 실험하였다.

본 논문에서 사용한 방법은 캐시안의 크고 작은 파일 간에 균형 잡힌 결과를 갖도록 지원하여 적중률과 바이트적중률에 있어 성능 개선의 효과가 높게 나타났다. 결과적으로는 멀티미디어 사용이 많은 웹 서버에서 더 큰 성능의 개선이 나타났으며, 적중률 개선도 중요하지만 바이트적중률의 개선 효과도 뚜렷이 나타났다. 이렇게 개선된 효과를 캐시 크기로 보면 통합 캐시 보다는 파일 타입별 캐시가 본 실험에서는 최소 1/3에서 최대 2/3 가까이 기억공간을 절약하면서 유사한 성능을 보이고 있다.

참 고 문 헌

[1] R. El Abdouni Khayari, R. Sadre, B. R. Haverkort, and M. Pistorius. A Class-Based Least-Recently Used Caching Algorithm for WWW Proxies. Technical report, Laborato-

ry for Performance Evaluation and Distributed Systems, RWTH Aachen, 2002.

[2] B. M. Duska, D. Marwood and M. J. Feely, "World-Wide-Web Client Proxy Caches", Proc. of the USENIX Symposium on Internet Tech. and System", DEC. 1997.

[3] Sambit Sahu, Prashant Shenoy, Don Towsley, Design Considerations for Intergrated Proxy Servers", Proc. of IEEE NOSSDAV '99, pp. 247-250, June 1999.

[4] Jia Wang.,"A survery of Web caching schemes for the Internet," ACM Computer Comunication Review, 29(5), pp. 36-46, October, 1999.

[5] Greg Barish and Katia Obraczka, "World Wide Web Caching: Trends and Techniques, "IEEE Communications Magazine Internet Technogy Series, May, 2000.

[6] Thoms Larkin,"An Evaluation of Caching Strategies for Clustered Web Servers", Master's thesis, Univ. of Dublin, September. 2001.

[7] Igor Tatarinov, "Cache Policies for Web Server". http://www.cs.ndsu.nodak.edu.tatarino/cache-policies.ps

[8] Edith Cohen and Haim Kaplan, "Exploiting Regularities in Web Traffic Patterns for Cache Replacement", In Proceedings of the 31st Annual ACM Symposium on Theory of Computing. ACM, 1999.

◐ 저 자 소 개 ◑

임 재 현

1986년 중앙대학교 전자계산학과 졸업(학사)
1988년 중앙대학교 대학원 전자계산학과 졸업(석사)
1998년 중앙대학교 대학원 컴퓨터공학과 졸업(박사)
1998~현재 공주대학교 멀티미디어정보·영상공학부 교수
관심분야 : 유비쿼터스, 상황인식, 인터넷 기술
E-mail : defacto@kongju.ac.kr