클러스터 Web 서버 상에서 히스토그램 균등화를 이용한 내용기반 부하분산 기법

김종근°, 최황규
강원대학교 전기전자정보통신공학부
godofdream@mail.kangwon.ac.kr, hkchoi@kangwon.ac.kr

Content-Aware Load Balancing Technique Based on Histogram Equalization in Web Server Clusters

Jong-Geun Kim, Hwang-Kyu Choi
Dept. of Electrical and Computer Engineering, Kangwon University

요 약

본 논문은 대용량 클러스터 기반의 웹 서버를 위한 새로운 내용 기반 부하 분산 기법을 제안한다. 제안된 기법은 웹 서버 로그의 URL 항목에 해시 함수를 적용하여 얻어지는 해시 값에 요청 빈도와 전송될 파일 크기를 누적하여 히스토그램을 생성한다. 그 결과로 생성된 히스토그램의 누적 분포에 히스토그램 균등화 변환함수를 적용하여 각각의 서버 노드에 해시 값에 따라 분포하는 부하를 균등하게 할당할 수 있다. 제안된 부하 분산 기법의 효율성 검증을 위한 시뮬레이션에서 히스토그램 균등화 기법은 서버의 지역적인 캐시 활용과 부하의 균등 분산 등의 장점으로 우수한 성능을 나타냄을 보인다.

1. 서 론

최근 인터넷 사용자의 급증으로 기존의 웹 서버 시스템들의 확장이 불가피하게 되었다. 그러나 기존의 대부분의 서버는 고성능의 단일 시스템으로 그 확장에는 한계가 있을 뿐 아니라 많은 비용을 필요로 한다. 이러한 문제 해결을 위해 여러 대의 고성능 PC 또는 워크스테이션을 고속의 네트워크로 결합하는 클러스터링 기법에 대한 많은 연구가 진행되고 있다. 이러한 클러스터링 기법은 확장성과 가격 대 성능이 우수한 웹 서버 시스템을 제공한다.

지난 몇 년간 클러스터 서버의 기술은 상업적인 솔루션뿐만 아니라 실험적인 연구에 의한 방법에서도 주목할만한 성장을 보여 왔다. 이러한 클러스터 서버 기술에서 가장 중요한 문제는 어떻게 클라이언트의 요청을 각 서버 노드에 균등하게 분배하느냐 하는 것이다.

전통적인 부하 분산 방식은 나름대로 우수한 부하 분산 성능으로 좋은 서버 확장성 및 클라이언트 투명성을 제공한다[1]. 그러나 최근 연구에서는 내용 기반(content-aware request distribution) 부하 분산 방법이 대두되고 있다.

기존의 부하 분산 기술이 부하 분산을 위해서 클러스터 내 서버 노드의 부하 정보(예: CPU 시간, 메모리 크기, 접속의 수 등)만을 이용하는 반면에, 내용 기반 부하 분산 방법은 내용(content)이나 서비스에 따라 요구를 분배한다. 이러한 기법은 특화된 서비스에 의한 서버 분할이 가능하게 하는 등 전체적인 서버 시스템의 성능 향상 및 유연성을 제공한다[2].

본 논문에서는 새로운 부하 분산 방법으로 히스토그램 균등화를 이용하는 부하 분산 기법을 제안한다. 히스토그램 균등화를 이용하는 부하 분산 기법에 의해 얻어지는 *[해시 값-서비스 노드]*의 설정을 주기적으로 갱신함으로써, 동적으로 클러스터 웹 서버 시스템의 부하를 고르게 분산할 수 있으며, 각 서버 노드의 캐시를 효율적으로 활용함으로서 전체적인 클러스터 시스템의 성능을 향상시킬 수 있다.

본 논문은 먼저 2장에서 최근 클러스터 웹 서버 시스템에서 중요 연구 과제인 내용 기반 부하 분산에 대한 연구를 알아보고, 3장에서는 히스토그램 균등화를 이용한 부하 분산 기법을 설명한다. 또한 제안된 기법의 성능 평가를 위하여 시뮬레이션 모델과 성능 분석 결과를 4장에서 기술하고, 마지막으로 5장에서 결론을 맺는다.

2. 내용 기반 부하 분산

전통적인 부하 분산 기술의 대부분은 웹 서버의 내용(content)에는 무관하게 서버 노드간의 부하 정보를 바탕으로 라운드 로빈 방식(round robin)이나 최소 연결 방식(least connection)으로 클라이언트 요청을 할당한다. 이와

달리 최근 몇 년간의 연구에서는 내용의 타입이나 서비스에 따라 클라이언트 요청을 분배하는 내용 기반 부하 분산(content-aware request distribution)을 통해 전체적인 클러스터 시스템의 확장성과 유연성을 제공하고 있다[2].

내용 기반 부하 분산에 의한 잠재적인 장점은 다음 세 가지로 생각할 수 있다. 첫째, 서버 노드의 메인 메모리 캐시의 적중률(hit rates) 향상으로 클러스터 시스템의 성능이 향상된다. 둘째, 서버 노드의 분할을 통해 2차 저장장치의 확장성을 향상시킬 수 있다. 셋째, 요청의 타입(예, 오디오, 비디오 등)에 따라 특정한 서버 노드를 배치 운영할 수 있다[2]. 지금까지의 내용 기반 부하 분산 방식은 주로 첫 번째 장점을 고려한 부하 분산 방식의 연구가 활발히 진행되고 있다.

Locality-aware request distribution(LARD)은 내용 기반 부하 방식으로 위에서 설명한 첫 번째 장점에 초점을 맞춘 방식이다[3]. 중앙 집중적인 방식의 서버 구조에서 front-end는 모든 클라이언트 요청을 back-end 노드에 새로운 연결을 hand off하고 데이터를 전달하는 역할을 한다. Front-end에서 모든 클라이언트와 back-end 노드의 연결을 관리하기 때문에 이러한 연결 정보를 바탕으로 손쉽게 하나의 요청을 동적으로 변화하는 back-end 노드의 부하 정보를 고려하여 서비스할 노드에 할당한다. 다시 말해서, 요청을 가장 최근에 처리한 노드 중에서 가장 접속이 적은 노드에 할당 하게된다. 전통적인 부하 분산 방식의 라운드 로빈(RR) 또는 서버 노드의 성능의 차이에 따른 가중치를 적용한 WRR와 달리 전체의 작업문서가 아닌 부분적인 작업문서를 메인 메모리에 캐시 함으로써 캐시의 적중률이 높아지는 것은 자명한 사실이며 캐시의 지역성을 효과적으로 활용하는 부하 방식인 LARD는 전체적인 시스템의 성능 성능향상을 가져온다.

HACC[4]와 FLEX[5] 시스템은 LARD와 유사한 방법으로, HACC는 윈도우즈 NT 기반에서 구현된 Smart Router를 통해 일부 문서의 그룹(예, 정적 문서와 동적 문서)을 서비스하는 노드에 요청을 분배한다. FLEX는 웹 호스팅 서비스를 위한 부하 분산 방식으로 웹 서버 로그를 수집, 트래픽 패턴을 분석하여 서버 노드에 웹 사이트를 균등하게 할당한다. 주기적으로 DNS 서버의 사이트-서버 할당의 설정만을 갱신함으로써 추가적인 비용이 없이 효율적인 부하 분산이 가능하다.

3. 히스토그램 균등화를 이용한 부하분산 기법

본 논문에서 고려하는 클러스터 웹 서버의 구조는 부하 분산을 위한 부하 분산기와 실제 요청을 처리하는 여러 개의 웹 서버 노드, 그리고 부하 분산을 위한 설정 테이블을 갱신하고 웹 서버의 부하 상태를 모니터하는 마스터 노드로 구성된다. 사용자의 요청은 인터넷을 통해 부하 분산기에 전달된다. 부하 분산기는 논문에서 제안된 부하 분산 기법에 의하여 미리 설정된 *[해시 값 - 서비스 노드]* 테이블에 의하여 요청을 웹 서버에 전송하는 역할을 수행한다. 클러스터 웹 서버 노드들의 설정 테이블의 갱신은 마스터 노드(Master Node)에서 수행된다. 마스터 노드는 각 웹 서버의 부하 상태를 모니터하고 주기적으로 또는 과부하 상황에 있는 웹 서버로부터 과부하 경고 피드백을 받았을 때, 각 웹 서버로부터 로그를 수집한다. 수집한 웹 로그의 분석은 제안하는 기법인 히스토그램 균등화를 이용한 부하분산 기법을 적용하며, 새로운 부하 할당 테이블로 각 웹 서버의 설정을 갱신하여 준다.

그림 1은 제안하는 기법인 히스토그램 균등화를 이용한 부하 분산 기법의 프레임워크를 보여준다. 히스토그램 균등화를 이용한 부하 분산 기법은 먼저 각 웹 서버로 웹 로그를 수집 분석한다. 웹 로그의 분석 결과 파일의 빈도와 크기를 누적한 해시 히스토그램을 생성하게 되며, 히스토그램에 균등화 변환함수를 적용함으로써 부하를 웹 서버에 고르게 할당할 수 있다.

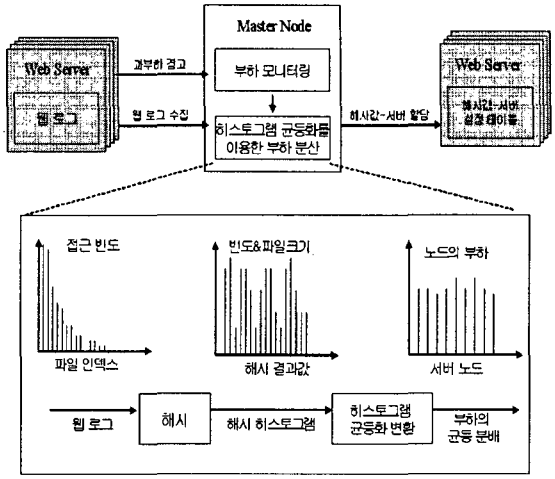

그림 1. 히스토그램 균등화를 이용한 부하 분산 기법의 프레임워크

단계 1) 웹 로그 수집 및 분석

웹 로그는 사용자가 요청한 파일과 파일에 포함된 여러 객체들에 대하여 하나의 레코드로 기록된다. 웹 로그의 하나의 레코드에서 소스 IP 주소는 요청 패킷을 보낸 클라이언트 주소이고, 파일에 접근한 날짜 및 시간 항목을 포함한다. 그리고 'GET', 'POST'와 같은 HTTP 메소드, 응답이 성공적으로 이루어 졌는지를 나타내는 응답 코드와 요청한 파일의 경로를 포함한 URL(또는 파일명)을 가지고 성공적으로 전송이 이루어졌을 경우 전송된 파일 크기가 포함된다.

웹 로그로부터 각 파일에 대한 접근 빈도와 파일 크기를 고려하여 해시 히스토그램을 생성하기 위해서, 성공적으로 응답한 로그 레코드의 URL 항목과 클라이언트로 전송된 파일크기 항목을 추출한다.

단계 2) 해시 히스토그램 생성

추출한 각 URL 항목에 해시 함수를 적용하여 얻어지는 해시 값에 전송된 파일 크기를 누적한다. 웹 서버의 부하는 파일에 대한 접근 빈도와 파일 크기에 따른 서버의 프로세싱 시간으로 결정된다. 그러므로 생성된 해시 히스토그램은 각 파일의 빈도에 가중치를 달리 적용하여 부하가 랜덤하게 해시 범위에 분포하게 된다.

단계 3) 히스토그램 균등화

웹 로그의 해시 히스토그램의 파일 빈도와 크기에 대한 부하를 각 서버 노드에 분할하기 위해서 히스토그램 균등화(histogram equalization) 기법을 사용한다. 먼저, 히스토그램 균등화 변환을 위해서 해시 히스토그램의 확률 분포를 구한다. 히스토그램의 확률 분포에 누적 분포함수(cumulative distribution function)를 이용하여 히스토그램의 확률 누적 분포를 구한다. 확률 누적 분포로부터 클러스터 서버 노드에 균등하게 해시 결과 값을 할당하기 위해서 다음의 변환함수를 이용한다.

$$y = \lceil P \sum_{m=0}^{x} H_{x(m)} \rceil \quad (1)$$

여기서, ⌈ ⌉는 상한(ceiling) 함수이고. P는 크기 벡터로서 서버 노드의 성능에 따라 부여된 정수의 가중치의 합이다. 즉, 동일한 성능을 가지는 경우, 각 서버의 가중치는 1로서 서버 노드의 개수와 동일하게 되며, 서버 노드의 성능이 다른 경우, 성능에 따라 다르게 부여된 가중치의 합이 된다. 해시 히스토그램으로부터 해시 범위 $0 \le m < x$에 대하여 변환함수를 적용하여 해시 값에 대응하는 서버 노드를 선택한다.

그림 2는 웹 로그의 해시 결과 값에 따른 파일의 접근 빈도와 크기를 누적한 해시 히스토그램의 확률 분포와 히스토그램 균등화 변환에 의한 부하 분산 결과 히스토그램의 예를 보여준다. 그림 2(a)와 같이 해시 결과 값에 대한 파일의 접근 빈도와 크기를 누적한 해시 히스토그램의 확률 분포가 나타날 때, 동일한 성능을 가지는 4개의 서버 노드에 부하를 균등하게 할당하기 위해서 식(1)의 히스토그램 균등화 변환 함수를 적용한다.

$$y_0 = \lceil 4 \times 0.07 \rceil = \lceil 0.28 \rceil = 1$$

다음 해시 값에 대하여도 동일하게 적용하면,

$$y_1 = 1,\ y_2 = 1,\ y_3 = 2,\ y_4 = 2,\ y_5 = 2,$$
$$y_6 = 2,\ y_7 = 3,\ y_8 = 3,\ y_9 = 3,\ y_{10} = 3,$$
$$y_{11} = 4,\ y_{12} = 4,\ y_{13} = 4,\ y_{14} = 4,\ y_{15} = 4.$$

이다.

히스토그램 균등화 변환에 의하여 4개의 서버 노드로 부하가 분배된다. 즉 추출된 경계 값 1, 2, 3, 4에 의하여 해시 값 0~2까지의 범위는 서버 노드 1에 할당되고, 3~6까지의 해시 값은 서버 노드 2, 7~10은 서버 노드 3, 11~15의 해시 값은 서버 노드 4에 할당된다(그림 2(b)).

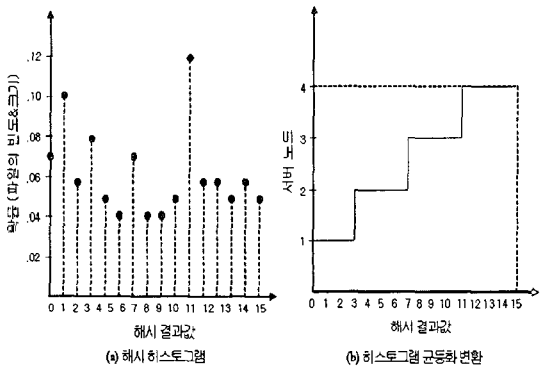

그림 2. 히스토그램 균등화 변환 예

기존의 내용 기반 부하 분산 방식(예, LARD)의 문제점은 동적으로 URL 테이블을 유지 관리하기 위해 많은 프로세싱 시간을 소비함으로 부하 분산기에서 부하 분산을 위한 오버헤드에 의하여 클러스터 서버의 확장성에 제약을 받는데 있다. 또한, 사용자의 요청에 대하여 빠르게 URL 테이블을 검색하기 위하여 다중 해시 테이블과 같은 구조를 필요로 하게 된다. 본 논문에서 제안하는 기법은 주기적으로 또는 과부하 상황에서만 웹 서버 로그를 분석하여 설정 테이블을 갱신함으로써 설정 테이블을 유지 관리하기 위한 오버헤드 문제를 해결할 수 있으며, 사용자 요청에 대하여 바로 해시 함수를 적용하여 요청을 분배하기 때문에 더욱 효율적이다.

4. 실험 및 성능 평가

본 논문에서 제안한 부하 분산 기법의 성능평가를 위하여 시뮬레이션을 수행하였다. 시뮬레이션 모델은 VC++를 사용하여 작성하였으며, 모델 파라미터는 [6][7]을 참조하였다. 시뮬레이션을 위한 포워딩(hand off) 오버헤드는 300 μsec를 사용하였다. [2]의 실험 값(약 300 μsec)에 근거한 수치이다. 서버 노드에서 요청을 처리하기 위한 비용은 메모리에 캐시된 파일의 요청을 처리하는 경우는 평균 파일 크기가 14.5KB일 때 초당 900개 정도의 요청을 서비스할 수 있으며, 디스크로 파일을 검색하는 경우는 초당 60개의 요청을 처리할 수 있다고 가정한다. 파일의 서비스 시간은 파일크기에 비례한다고 가정한다. 또한 서버 노드의 캐시 정책은 일반적인 LRU이며, 실험을 단순화하기 위하여 파일 단위로 캐시된다고 가정한다. 웹 로그는 웹 서버의 작업 부하 특성(SPECweb96)을 고려하여 임의 생성한 로그와 실제 웹 로그를 사용하였다. 시뮬레이션의 결과 각각의 부하 분산 방식(RR, LARD)과의 비교 평가를 위하여 동일한 임의 생성한 웹 로그를 요청으로 사용하며 웹 로그 파일 전체를 수행했을 때의 서버 노드당 처리량(request/sec)과 전송량(MB)을 비교하였다.

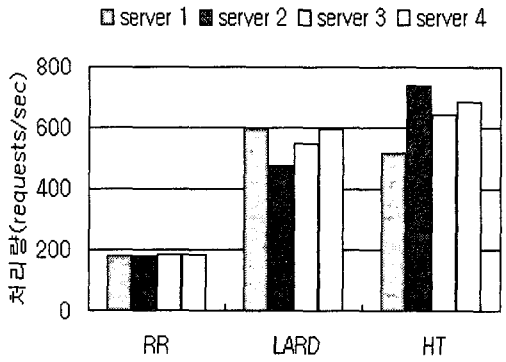

그림 3. 처리량 비교

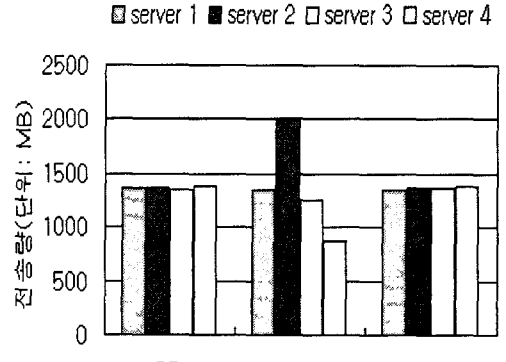

그림 4. 전송량 비교

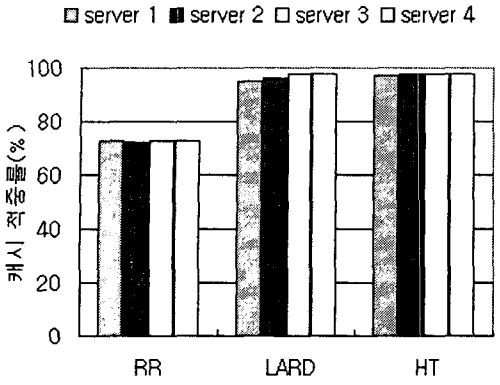

그림 5. 캐쉬 적중률 비교

그림 3, 4, 5는 제안하는 기법(HT)의 부하 분산 성능을 비교하기 위한 기존의 라운드 로빈(RR), LARD 방식의 서버 노드의 부하를 측정한 결과이다. 그림 3에서와 같이 특정 파일(content)에 대하여 서비스하는 노드를 제약하는 방식인 LARD와 HT가 전통적인 방식의 RR에 비하여 디스크의 접근을 줄이고 캐시 적중률을 향상시키는 장점에 의하여 처리량이 월등히 우수함을 알 수 있다. 그림 4, 5에서는 서버 노드당 부하 분산 상태를 보여준다. RR과 HT이 LARD 방식에 비하여 부하가 각 노드에 고르게 분산되고 있음을 알 수 있다. 따라서 제안하는 기법(HT)은 캐시의 지역성을 높이고, 부하의 고른 분산을 위한 효율적인 부하 분산 기법임을 알 수 있다.

5. 결 론

본 논문은 클러스터 웹 서버 상에서 부하의 균등 분산을 위한 내용 기반 부하 분산 기법의 성능 개선을 위한 새로운 히스토그램 균등화 변환 기법을 제안하였다. 제안된 기법은 서버 노드의 캐시 효율을 향상시키면서 부하를 균등 분산시키는 기법으로써 동적 부하 분산을 위해 주기적으로 웹 로그를 수집하여 분석한다. 하루에도 몇 백만의 요청을 받는 웹 서버의 방대한 로그를 수집하여 간결하게 각 로그 엔트리의 URL 항목에 해시 함수를 적용하여 얻은 파일의 요청 빈도와 크기를 누적하여 히스토그램을 생성할 수 있다. 그리고 웹 로그로부터 얻은 해시 히스토그램의 균등화 변환함수에 의해 서버 노드에 파일을 할당한다. 웹 로그의 간결한 분석 절차는 변화하는 요청의 패턴에 따라 동적으로 대응할 수 있게 함으로써 클러스터 웹 서버 시스템의 부하를 고르게 분산할 수 있으며, 각 서버 노드의 캐시를 효율적으로 활용함으로서 전체적인 클러스터 시스템의 성능을 향상시킬 수 있다.

참고 문헌

[1] V. Cardellini, M. Colajanni, and P. S. Yu, "Dynamic Load Balancing on Web Server Systems," IEEE Internet Computing 3(3), 28-39 1999.

[2] M. Aron, D. Sanders, P Druschel, and W. Zwaenepoel, "Scalable Content-aware Request Distribution in Cluster-based Network Servers," In Proc. of USENIX2000 Technical Conference, June 2000.

[3] V. S. Pai, M. Aron, G. Banga, M. Svendsen, P. Druschel, W. Zwaenepoel, and E. Nahum, "Locality-Aware Request Distribution in Cluster-based Network Servers," In Proc. of the 8th Conference on Architectural Support for Programming Languages and Operating System, San Jose, CA, Oct. 1998.

[4] X. Zhang, M. Barrientos, J. B. Chen, and M. Seltzer, "HACC: An Architecture for Cluster-Based Web Servers," In Proc. of the 3rd USENIX Windows NT Symposium, Seattle, WA, July 1999.

[5] L. Cherkasova, "FLEX: Load Balancing and Management Strategy for Scalable Web Hosting Service," In Proc. of the Fifth International Symposium on Computers and Communications(ISCC00), Antibes, France, July 3-7, 2000.

[6] R. Bianchini and E. V. Carrera, "Analytical and Experimental Evaluation of Cluster-Based Network Servers," World Wide Web Journal, 3(4), Dec. 2000.

[7] L. Cherkasova and M. Karlsson, "Scalable Web Server Cluster Design with Workload-Aware Request Distribution Strategy WARD," In Proc. of the 3rd International Workshop on Advanced Issues of E-Commerce and Web-Based Information System, June 2001.