한국시뮬레이션학회 논문지
Vol. 22, No. 3, pp. 71-79 (2013. 9)
http://dx.doi.org/10.9709/JKSS.2013.22.3.071

대규모 웹서버 클러스터 시스템의 운영방안 연구

박진원[1†]

Operational Scheme for Large Scale Web Server Cluster Systems

Jin-Won Park

ABSTRACT

Web server cluster systems are widely used, where a large number of PC level servers are interconnected via network. This paper focuses on forecasting an appropriate number of web servers which can serve four different classes of user requests, simple web page viewing, knowledge query, motion picture viewing and motion picture uploading. Two ways of serving different classes of web service requests are considered, commonly used web servers and service dedicated web servers. Computer simulation experiments are performed in order to find a good way of allocating web servers among different classes of web service requests, maintaining certain levels of resource utilization and response time.

Key words : Web Server Cluster System, Capacity Planning, Computer Simulation Experiment

요 약

PC급 성능의 서버를 네트워크로 연결하여 대규모 웹서비스에 사용하는 웹서버 클러스터 시스템이 널리 활용되고 있다. 본 논문은 단순 페이지 뷰, 지식 탐색, 동영상 뷰 및 동영상 업로드 등 4가지 형태의 웹 서비스를 제공하는 대규모 웹서버 클러스터 시스템을 대상으로 공동 사용 방식과 전용 사용 방식을 적용할 경우 각각 필요한 웹서버 규모를 예측해 본다. 이를 위해 일정한 수준의 자원 활용률을 유지하면서 응답시간을 짧게 유지하는 서버 배치 방안을 컴퓨터 시뮬레이션 실험을 통해 모색해 본다.

주요어 : 웹서버 클러스터 시스템, 용량 계획, 컴퓨터 시뮬레이션 실험

1. 서 론

최근의 컴퓨팅 환경이 클라우드 컴퓨팅, 네트워크 컴퓨팅, 클러스터링 등으로 변화하면서, 특히 주목을 받고 있는 기술이 대규모 웹서버 클러스터링(Web Server Clustering) 기술이다. 지난 1990년대 이래 개인용 컴퓨터가 네트워크로 연결되면서 컴퓨터의 주 기능이 더 이상 계산 중심이 아니라 통신 및 정보 검색 위주로 개편됨에 따라, 대용량 데이터베이스를 기반으로 사용자가 원하는 데이터를 제공해주는 포털 서비스가 활성화되었다. 이러한 컴퓨팅 환경의 변화는 컴퓨터 시스템의 구성 자체에도 큰 변화를 가져오게 되었다.

초기의 컴퓨터 시스템은 컴퓨터 자체의 성능을 향상시키는 방향으로 사용자의 요구를 만족시키려 노력했다면, 컴퓨터 시스템 자체의 성능 향상이 어느 정도 한계에 부딪힌 2000년대에는 인터넷의 보편화와 더불어 수만 대에서 수백만 대에 이르는 소형 컴퓨터들을 네트워크로 연결하여 슈퍼컴퓨터 이상의 성능을 발휘하는 클러스터 시스템을 구성하는 형태로 변화하였다. 이는 단일 컴퓨터의 계산 능력보다는 대규모 데이터베이스에서 사용자가 원하는 데이터를 병렬로 검색하는 능력이 더 필요하게 된

* 이 논문은 2013학년도 홍익대학교 학술진흥연구비에 의하여 지원되었음.
접수일(2013년 7월 29일), 심사일(2013년 9월 12일), 게재 확정일(2013년 9월 17일)
[1] 홍익대학교 게임학부 게임소프트웨어전공
주 저 자 : 박진원
교신저자 : 박진원
E-mail; jinon@hongik.ac.kr

상황에 기인한다. 이렇게 구성된 클러스터 시스템은 주로 웹 서비스를 담당하게 되어 최근의 연구가 웹서버 클러스터를 대상으로 성능을 향상시키는 방향으로 이루어지고 있다.

웹서버 클러스터는 구성 방식에 따라 여러 형태를 갖고 있다. Schroeder et. al.[1]은 사용자의 요구를 웹서버를 구성하고 있는 개별 시스템에 분배하는 방식에 따라 3가지 형태의 클러스터 시스템으로 구분했다. 즉, L4 스위칭과 L2 패킷 포워딩(L4/2), L4 스위칭과 L3 패킷 포워딩(L4/3) 그리고 L7 스위칭(L7) 클러스터링 시스템으로 분류하였다. 웹서버 클러스터 시스템의 성능은 주로 서버들 간의 작업부하 균등 분배에 초점이 맞춰져 연구되어 왔다. Choi[2]는 웹서버 클러스터 시스템의 작업부하를 RR(Round Robin), LC(Least Connection), WLC(Weighted Least Connection) 등의 작업부하 분배 방식의 시스템 성능을 실제 시스템 구축과 실험을 통해 제시했다.

부하분산을 이용하여 웹서버 클러스터 시스템의 반응시간을 단축하려는 소프트웨어적인 방법을 제시한 김석찬, 이영[3]에서는 서버 숫자가 최대 8개까지의 클러스터 시스템을 대상으로 기존의 하드웨어적 부하분산 방식에 대한 대안으로 소프트웨어 방식에 의한 부하 분산 방식을 제안하고 실험을 통해 이에 대한 성능평가를 수행하였다. 사용자의 요구내용을 기반으로 웹서버 클러스터 시스템을 구성하는 서버들에게 작업을 할당하는 내용기반 매핑 방식은 실험을 통해 최대 15.7%까지의 지연시간 기준의 성능을 향상시킬 수 있다는 연구도 발표되었다[4].

Pacifici 등[5]은 웹서버 클러스터 시스템에 대한 본격적인 성능 문제를 제기했다. 사용자 요구를 우선순위 사용자와 기본 사용자로 분류하여 동적으로 서버를 할당하는데, 서버에 과부하가 걸리지 않고 시스템의 전체 처리량이 극대화되고 사용자 요구에 대한 평균 시스템 반응시간이 최소가 되는 방향의 목적함수를 극대화하는 방안을 제시하여, 사용자 수가 5~20명일 경우를 대상으로 실험하여 그 결과를 제시했다. 정지영과 김성수[6]는 웹서버 클러스터 시스템을 대상으로 캐시 적중과 서버의 작업부하 상태를 고려한 효율적인 내용기반 부하분배 알고리즘을 제안하고 시스템 처리량을 성능평가 기준으로 설정하여 서버의 수가 1~10개인 시스템을 대상으로 실험하였다. 그리고 L4 방식과 L7 방식의 부하분배 방식을 비교하기도 했다.

Chen과 Chen[7]은 사용자 요구를 내용에 따라 분류하고 서버에 작업부하를 분배할 때 특별히 고안된 비용함수를 이용하는 알고리즘을 제안했다. 즉, 사용자의 요구를 정적인 서비스, 동적인 서비스 및 멀티미디어 서비스로 분류하고 각각의 서비스에 알맞는 서버를 선택하여 작업을 분배하는 방식을 제안했다. 그러나 이 연구는 호스트(서버)의 수를 2개, 4개, 8개로 제한하여 웹서버 클러스터 시스템의 규모면에서 한계를 보인다. 그 외 동적인 부하분산 방식을 제안한 논문으로 주기적인 부하 가중치 산정을 통해 시스템의 부하를 감지하여 이를 다음 주기의 부하 분산에 활용하는 방식[8]과 비 주기적인 부하 갱신을 통한 부하 분산 기법을 제안한 방식이 발표되었다[9].

사용자의 요구는 웹서버와 응용서버가 순차적으로 처리하게 되는데 대부분의 부하분산 관련 연구가 웹서버에 국한되어 있으나 웹서버와 응용서버는 사용자 요구의 분포가 다르다고 주장하여, 웹서버 1, 2, 5개와 응용서버 10, 20, 30개를 대상으로 전체적인 시스템 처리율, 자원 사용률, 시스템의 반응 시간 등이 향상되는 응용서버 중심 부하분산 방식을 제안하고 이에 대해 실험을 통해 분석한 연구결과도 발표되었다[10].

유한 버퍼를 갖는 2개 서버 시스템에 대해 중앙집중식 작업부하 분산방식과 분산식 작업부하 분산방식의 성능을 해석적 방법에 의해 분석한 사례가 있다[11]. 2개 서버 시스템에 대해 큐잉 모델을 이용하여 분석한 것인데 두 가지 부하 분산 방식이 작업부하 정도가 낮을 때는 성능에 큰 차이가 없으나 작업부하가 높은 수준일 때는 상당한 성능 차이를 보이는 것으로 분석되었다. 또한, 장혜천 등은[12] 웹서버 클러스터에서 서버 수의 증가가 성능향상에 미치는 영향을 분석했다. 우선 소규모 웹서버 클러스터 시스템을 구성하여 성능을 분석한 후 이를 확장하여 대규모 웹서버 클러스터의 성능 예측에 활용하였다. 이와 같이 기존의 웹서버 클러스터 시스템에 대한 성능 분석은 실제 시스템을 구축하여 실험을 통해 동적 부하 분배 알고리즘의 성능을 분석하는데 초점을 맞추고 있다. 즉, 실제 시스템을 구성하여 실험해야 하기 때문에 시스템이 소규모에 국한되고, 동적 부하 분산 알고리즘을 시험하기 위해 가상의 작업 부하를 가정하고 있다.

본 논문은 실제 시스템을 구성하지 않고 컴퓨터 시뮬레이션을 이용하여 대규모 웹서버 클러스터 시스템을 대상으로 실험하였으며 실제 웹서비스를 수행하고 있는 사이트의 사용자 이용 현황을 기반으로 작업 부하를 추정하고 이를 실험에 반영하였다. 추정한 작업 부하를 기반으로 대규모 웹서버 클러스터 시스템을 컴퓨터 시뮬레이션 프로그램 상에 구축하고 서로 다른 형태의 사용자 서비스 요구를 처리하기 위해 웹서버들을 어떻게 배치하고 운영해야 하는가 모색하는 것이 본 논문의 목적이다.

2. 성능평가 모델링

본 논문은 수백 대 이상 수만 대로 구성된 웹서버 클러스터 시스템을 효율적으로 운영하기 위한 방안을 연구하는 것이다. 즉, 대규모 웹서버 클러스터 시스템을 사용자가 기대하는 성능을 발휘하기 위해 어떻게 운용해야 하는가에 초점을 맞추고 있다. 대규모 웹서버 클러스터 시스템의 성능을 분석하는데 사전에 설정하거나 특성을 분석해야 할 시스템 요소는 세 가지로 분류해 볼 수 있다. 즉, 사용자 혹은 시스템 운영자가 기대하는 성능 지표, 사용자의 요구 작업 유형과 각 유형별 작업 부하, 그리고 웹 클러스터 시스템 자체의 동작 특성 등이다.

웹서버 클러스터 시스템에 대해 사용자 혹은 시스템 운영자가 기대하는 성능 지표는 두 가지 측면에서 살펴볼 수 있다. 사용자 입장에서는 시스템 반응시간(혹은 시스템 체류시간)이 주 관심사이고 시스템 운영자 입장에서는 자원 사용률과 부하 분배의 합리성 등이다. 앞 절에서 살펴본 바와 같이 기존 연구는 소규모 웹서버 클러스터 시스템을 대상으로 동적 부하분산 알고리즘을 위주로 분석했고 대용량, 대규모 웹서버 클러스터 시스템을 대상으로 최적 운영 방안에 대한 논의는 제한적이었다. 이는 연구 방법이 실제 시스템을 구축하는 실험을 통해 이루어졌고, 작업 부하도 소수의 사용자를 가정한 상태에서 수행되었기 때문이다. 그러나 대규모 웹서버 클러스터 시스템의 운영에서는 부하 분산에 관한 이슈 뿐 만 아니라 사용자가 경험하는 시스템 반응시간이 더 중요한 이슈인 것으로 보인다[13]. 따라서 본 논문에서는 사용자 측면에서 시스템 반응시간을 최소화하고, 시스템 운영자 입장에서 안정적인 시스템 운영을 위해 필요한 웹서버 규모를 설정하는 문제와 각 사용자 요구마다 몇 대의 웹서버를 배치하는 것이 전체 시스템 차원에서 효율적인가를 결정하는 문제에 초점을 맞출 것이다.

2.1 사용자가 요구하는 작업의 유형과 각 유형별 작업 부하

사용자의 작업 부하와 관련하여 수집된 1차 자료는 국내에서 운영되는 대표적인 포털 업체인 Naver를 중심으로 Google의 자료를 일부 활용한 것이다. 2007년 Naver를 기준으로 1일 방문자 수는 총 2,200만 명으로 발표되었다[14].

Table 1은 사용자의 요구 유형별 분류 및 빈도를 나타낸 것으로, 페이지 뷰가 대부분을 차지하고 있고 지식 검색은 페이지 뷰의 약 1/9 정도의 빈도를 보이고 있으며, 동영상 뷰(MPC, Motion Picture Contents Viewing)나 동영상 업로드(UPLD, UPLoaDing)는 상대적으로 빈도가 매우 낮음을 알 수 있다. 컴퓨터 시뮬레이션 모델에서는 각 유형별 도착시간 간격에 대한 분포를 지수분포로 가정하였다.

Table 1. Types and Frequencies of User Requests

User Request Types	Visits per day	Ave. Arrival Interval
1. Page View	1 billion	0.0864ms
2. Query	0.11 billion	0.7854ms
3. MPC	2 millions	43.2ms
4. UPLD	20 thousands	216.0ms

2.2 웹서버 클러스터 시스템의 동작 매커니즘

웹서버 클러스터 시스템의 동작은 복잡하다. 시스템 외부와는 인터넷으로 연결되어 있고, 내부도 네트워크로 연결되어 있다. 또한 개별 웹서버의 하드웨어 뿐 만 아니라 소프트웨어와 동작 방식도 전체 시스템 성능에 큰 영향을 미친다. 웹서버 클러스터 시스템의 서비스 품질은 하드웨어 측면에서 다음 두 가지 요소에 의해 좌우된다. 첫째, 네트워크의 자료 전송 속도이다. 이는 인터넷 link의 bandwidth에 따라 결정되는 요소이다. 둘째, 하드웨어 구성 요소의 성능이다. 즉, CPU, RAM 및 I/O 성능 등 각 서버 구성 요소의 성능에 따라 결정되는 요소이다.

웹서버 클러스터 시스템의 서비스 품질을 소프트웨어, 혹은 운영 측면에서 향상시키려는 노력이 계속 연구되고 있다. 소프트웨어 측면에서 시도되고 있는 성능 향상 방안은 Load Balancing Strategy, DNS (Domain Name Server) 사용의 최소화, Core 사용, Replica 사용(Google, Hadoop 등)이 시도되고 있다. 이러한 성능향상 방안에 대한 고찰과 더불어 박진원[15]은 국내 연구기관에서 개발한 웹서버 클러스터 시스템을 대상으로 사용자에 대한 전체 서비스 소요 시간의 구성을 컴퓨터 시뮬레이션 모델을 이용하여 분석하여, 웹서버 처리시간이 네트워크 전송시간보다 많이 소요되고 있음을 보여주었다.

웹서버 클러스터 시스템을 대상으로 컴퓨터 시뮬레이션 모델을 구축하는 과정에서 각 서버는, Core 개념을 도입하여 사용자 파일 서비스 요구의 95%는 Local에서 처리하고 5%만이 redirecting을 통해 다른 웹서버에 의해 처리된다고 가정했다. 또한, 네트워크는 트리 형태로 구성되어 있으며 DNS 서버는 Internet Gate에 1대 혹은 그 이상 설치되어 있다고 설정했다. 다음은 지금까지 발표된

논문들을 기반으로 설정한 컴퓨터 시뮬레이션 모델의 시스템 파라미터들이다.

- 내부 DNS (Domain Name Server)가 Real IP Address를 변환하는데 소요되는 TCP Communication Overhead setup time: 75ms (Olshefski[16] 참조).
- 최초 방문자를 위한 인증시간(Authentication time): 20ms (Chung[17] 참조).
- 10Gbps의 용량을 갖는 네트워크: 4.8~6.0 Gbps 정도의 유효 전송률[18].
- HTTP 처리 시간: 300ms[16].
- 자료 검색 시간: 평균 400ms, 삼각분포 [200, 400, 600] 가정 (Zhu[19] 참조).
- 메모리 접근시간: Cardellini[20]에 따라, 1MB 데이터를 메모리에서 읽을 때 10ms, 디스크에서 읽을 때 100ms가 소요되며, 쓰기에는 이의 2배 시간이 소요된다는 사실을 근거로 계산. 메모리에서 직접 데이터를 가져 올 확률을 95%로 가정하고 디스크에서 데이터를 가져 올 확률을 5%로 가정하여 계산한 결과, 표준 크기인 16.125KB 데이터를 읽어 오는 시간은 (16.125*10/1,000)*0.95 + (16.625*100/1,000)*0.05 = 0.2338ms. 동영상 자료를 읽는 경우(MPC)나 쓰는 경우(UPLD)에는 Memory Read와 데이터 전송이 최초의 1.5 KB를 읽은 다음에 pipelining 된다는 가정에서 최초의 1.5KB(read 단위) read 시간, 혹은 write 시간을 감안한 후 별도의 메모리 접근 시간은 산정하지 않음. 읽기: 1.5*10/1,000 = 0.015ms, 쓰기: 1.5*20/1,000 = 0.030ms.
- MPC(동영상 뷰), UPLD(동영상 업로드): Portal에서는 대개 동영상 파일 크기에 제한을 둔다는 전제 아래 동영상 데이터는 16MB~80MB 크기의 일양분포(Uniform Distribution)를 갖는다고 가정. 인터넷에서 통용되는 동영상 파일 포맷을 기준으로 1분~5분에 해당하는 데이터 크기를 설정한 것임.

2.3 수학적 모델링 및 분석의 한계

웹서버에 도착하는 사용자들의 도착 시간 간격을 지수분포로 가정하는 것은 일견 타당해 보인다. 그러나 인터넷 서비스의 특성상 일정 시간대에 사용자의 요구가 몰리는 Peak Time이 존재할 것으로 예상되고, 밤늦은 시간이나 새벽에는 아무래도 사용자 요구가 한산할 것으로 보인다. 따라서 하루 전 시간대를 통해 일률적으로 사용자 도착 시간 간격에 대한 분포를 지수분포로 가정하는 것은 무리가 있어 보인다. 그러나 본 논문에서는 분석의 편의를 위해, 사용자 도착시간 간격에 대한 분포에서 Peak Time은 고려하지 않았다. 또한 시스템 구성 요소들의 서비스 시간에 대한 분포는 다단계 네트워크를 통한 데이터의 이동과 다양한 컴퓨팅 요소의 동작으로 이루어지는 시스템의 속성상 어느 하나의 특정한 분포로 가정할 수 없다.

사용자가 예상하는 시스템 반응시간은 사용자 요구의 빈도, 시스템의 서비스 방식 및 네트워크 요소와 컴퓨팅 요소들의 서비스 시간에 의해 좌우될 것이다. 그러나 이러한 요소들의 특성을 해석적으로(analytically) 분석하는 것은 사실상 불가능해 보여 컴퓨터 시뮬레이션 모델을 통해 분석해 보고자 한다.

3. 컴퓨터 시뮬레이션 모델 및 실험, 결과 분석

본 논문에서 분석의 모델로 설정한 웹서버 클러스터 시스템은 국내 정부출연연구소에서 개발한 GLORY 시스템이다. GLORY 시스템에 대한 구체적인 내용은 박진원[15]에 서술되어 있고, 단일 데이터센터 내 클러스터 시스템에 대한 하드웨어 구성은 Fig. 1과 같다.

Fig. 1은 실제로 구성 가능한 GLORY의 최소 규모를 보여주고 있다. GLORY 시스템은 2,048대 규모의 단일 데이터센터용 웹서버 클러스터가 네트워크로 연결되어 수백만 대까지 확장할 수 있는 시스템이 개발되어 현재 사용 중이다. 본 연구에서는 실험의 편의를 위해 실제 사용자 요구의 1/10 수준을 가정하여 실험을 진행하였다.

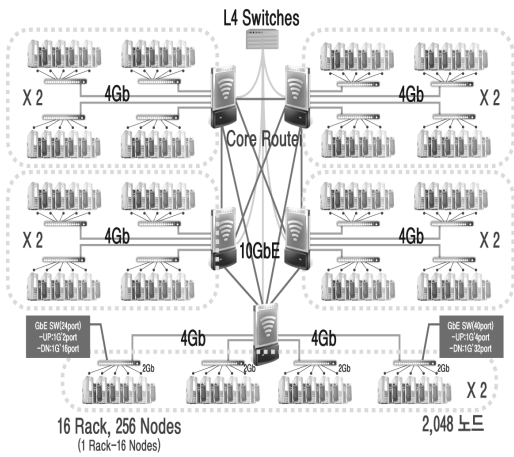

Fig. 1. Hardware for Clusters in a Data Center

3.1 컴퓨터 시뮬레이션 모델

2.1에서 살펴 본 작업 부하 자료와 2.2에서 설명한 웹서버 클러스터 시스템의 동작 매커니즘을 분석해 보면 다음과 같은 실험 대상 웹서버 클러스터 시스템의 특성을 알 수 있다. 첫째, 동영상 뷰와 동영상 업로드 요구는 페이지 뷰와 지식 검색 요구에 비해 현저하게 요구 빈도가 낮아 전체 웹서버 클러스터 시스템의 성능에 큰 영향을 미치지 못한다. 따라서, 이들 서비스를 위한 서버를 별도로 배치하여 페이지 뷰와 지식 검색을 위한 서버와 분리하는 것이 타당해 보인다. 둘째, 페이지 뷰 서비스의 도착 시간 간격($1/\lambda_1$)은 평균 0.864ms, 이를 위한 서비스 시간($1/\mu_1$)은 평균 317.5ms이고, 지식 검색에 대한 사용자 도착 시간 간격($1/\lambda_2$)은 평균 7.854ms, 이를 위한 서비스 시간($1/\mu_2$)은 평균 700.7ms이다. 따라서 교통밀집도 ρ가 1보다 작게 하려면 c_1, c_2(페이지 뷰와 지식 검색용 웹서버 대수)가 각각 368, 90 이상이어야 함을 알 수 있다. 따라서 전체 웹서버 수는 적어도 460대 이상이어야 하므로 여유분을 감안하여 그 이상의 수준에서 실험을 시행해야 할 것이다. 셋째, 이전 실험에서[15] 나타난 바와 같이 DNS 대수가 전체 시스템 성능에 큰 영향을 미치고 있어 적절한 수준의 DNS 대수를 배치해야 한다.

분석 대상 웹서버 클러스터 시스템에 대한 컴퓨터 시뮬레이션 모델은 Arena를 이용하여 구축되었다. 본 시뮬레이션은 사용자에서 서비스 요구가 시작되어 사용자에게 최종적으로 결과가 전송되는 전 과정을 다룬 것이 아니다. 이는, 사용자의 인터넷 사용 행태, 인터넷의 구조 및 특성 등을 감안해야 하는 어려운 점이 있고 본 시뮬레이션의 목적이 웹서버 클러스터 시스템 자체의 성능을 분석하는 것이므로 모델링의 범위를 웹서버 클러스터 시스템 자체에 국한시켰다. 즉, 인터넷을 통해 입력된 사용자 요구 내용이 웹서버 클러스터 시스템에 도착한 시점부터 웹서버 클러스터 시스템이 서비스를 마치고 인터넷에 그 결과를 띄워 보내는 순간까지를 대상으로 하였다.

3.2 실험 및 결과 분석

전체 웹서버 클러스터 시스템의 성능은 각 사용자 요구별 응답시간을 사용자 서비스 요구 도착율에 따라 가중평균한 평균 시스템 응답시간과 DNS와 웹서버들의 활용률 등으로 설정하고, 모든 서비스를 모든 웹서버가 처리할 수 있는 공동 사용(Pooling) 시스템을 가정한 500대의 웹서버 클러스터 시스템을 대상으로 첫 단계에서 실험하고 그 결과를 Table 2에 나타냈다.

Table 2는 DNS 대수가 전체 웹서버 클러스터 시스템 성능에 큰 영향을 미치고 있음을 보여주고 있다. 따라서 제2단계 실험에서는 DNS를 일단 3대 배치하고 웹서버 대수를 증가시키면 DNS 활용률과 웹서버 활용률이 어떻게 변하는가 관찰하였다. 앞의 실험에서 나타난 바와 같이 Table 1의 사용자 요구 도착율을 기준으로 DNS는 3대를 배치하고 Pooling 시스템을 가정하여 웹서버 대수를 450대에서 시작하여 점차 증가시키면서 평균 응답시간과 웹서버 활용률을 살펴보았다.

Table 3에 보인 실험은 Table 2의 실험 결과를 반영하여 좀 더 긴 평형상태 도달 시간을 설정했다. 즉, 대략 11,000ms 이후 시스템의 평균 응답 시간이 계속 변하다가 61,000ms에 이르면 비교적 안정적인 상태를 보인다는 점과, 그 후 111,000ms까지 평균 응답시간의 변화가 거의 없다는 점을 감안하여, 자료 수집 시작 시각을 11,000ms로, 시뮬레이션 종료 시각을 61,000ms로 설정하였다. Table 3에서 보는 바와 같이 웹서버 대수가 650대를 넘어서면 평균 응답시간 358.44ms와 DNS 활용률 0.7559에서 변화가 없고 웹서버 활용률만 0.7288에서 0.5921로

Table 2. Experiments for Pooling System (500 Web Servers, 1 DNS, * 2 DNS, ** 3 DNS)

Data collect Start time(ms)	Data collect End time(ms)	Ave. Response time(ms)	DNS Utilization	Web Server Utilization
1,000	11,000	383.67	1.0000	0.9187
1,000	21,000	404.58	1.0000	0.9210
1,000	31,000	427.54	1.0000	0.9161
1,000	61,000	516.94	1.0000	0.9149
1,000	61,000	415.95	1.0000*	0.9302
1,000	61,000	359.09	0.6861**	0.9306
1,000	111,000	358.81	0.7203**	0.9278

지속적으로 감소하는 것을 알 수 있다.

다음 제3단계에서는 일정 수준의 자원 활용률(예: 0.55 이하)을 유지하기 위해 웹서버를 얼마만큼이나 배치해야 하는가를 결정하기 위한 실험을 실시하였다. 이전 실험(Table 3)에서 나타난 바와 같이 DNS는 최소 3대 이상 배치해야 하고 웹서버도 800대 이상 배치해야 웹서버 활용률이 0.60 이하를 유지하는 것을 알 수 있다. 이번 실험에서는 MPC(동영상 뷰)와 UPLD(동영상 업로드)를 위한 전용 서버를 배치하기로 했다. 이는 차후에 실시할 각 서비스 유형별로 웹서버를 배치하는 전용 시스템(Dedicated System)에 대한 예비 실험을 하기 위함이다.

Table 4는 웹서버 대수를 변화시키는데 따르는 평균 응답시간, DNS, 웹서버, MPC서버, UPLD서버 등의 활용률 변화를 보여주고 있다. 여기서 모든 웹서버의 활용률이 0.55 이하로 유지되려면 각각 몇 대씩의 웹서버를 배치해야 하는가 결정하는 것이 목적이다. Table 3에서 보인 실험 결과, 우선 웹서버(페이지뷰 와 검색을 위한 서버)를 800대로 설정하고 DNS는 3대로 시작하였다. 또한 MPC 서버는 3대, UPLD 서버는 1대로 시작하였는데 이는 MPC 서비스의 평균 도착 시간 간격이 432ms이고 평균 서비스 시간이 1,250.5ms이어서 교통밀도(traffic intensity, $\rho = \lambda/c\mu$)가 1보다 작게 되기 위한 최소한의 c값인 3으로 정의한 것이다. 여기서 UPLD 서비스는 평균 도착시간 간격이 2,160ms이고 평균 서비스 시간이 1,236.0ms이므로 c값을 1로 설정하였다. 여기서 읽기 시간이 쓰기 시간보다 더 오래 소요되는 것은 2.2절에서 설명한 바와 같이 MPC 서비스는 메모리에서 95%, 디스크에서 5% 읽어 오는 것으로 가정했으나 UPLD 서비스는 사용자 요구가 메모리에 쓰는 동작까지 만을 포함하고 있기 때문이다.

Table 4에서 웹서버 배치를 (800, 3, 3, 1)로 설정할 경우 DNS 활용률이 0.7140, MPC 서버 활용률이 1.0으로 나타나 DNS와 MPC 서버를 우선 더 배치해야 함을 알 수 있다. 결국, DNS 4대, MPC 서버 5대를 배치해야 그들의 평균 활용률이 0.55 이하로 감소함을 알 수 있다. 웹서버 배치(800, 4, 5, 1)에 대해 시스템이 평형상태에 도달해 있는가를 검증하기 위해 장시간의(11,000～111,000) 실험을 실시해 보았으나 실험 결과는 거의 변화가 없어 이전 실험 시간을 유지하기로 했다.

다음에는 웹서버의 평균 활용률을 0.55 이하로 유지하기 위한 적절한 웹서버 수를 결정하기 위해 웹서버 수를 840, 860으로 증가시켜 실험을 실시하였더니 860일 때 웹서버 활용률이 0.55 이하로 내려갔고, 웹서버 수를 900대

Table 3. Performance Changes with Web Servers

No. of Web Servers	Ave. Response time(ms)	DNS Utilization(%)	Web Server Utilization (%)
450	6579.93	0.7688	1.0000
500	360.40	0.7518	0.9462
550	358.76	0.7533	0.8646
600	358.44	0.7599	0.7895
650	358.44	0.7559	0.7288
700	358.44	0.7559	0.6767
800	358.44	0.7559	0.5921

Table 4. Web Server Allocation for Keeping Web Server Utilization

No. of Web Servers*	Ave. Response Time(ms)	DNS Utilization	Web Server Utilization	MPC Server Utilization	UPLD Server Utilization
800,3,3,1	363.67	0.7140	0.5876	1.0000	0.3324
800,3,4,1	357.87	0.7472	0.5906	0.6802	0.3623
800,4,4,1	358.74	0.5075	0.5878	0.7653	0.4410
800,4,5,1	358.16	0.5428	0.5897	0.5419	0.5406
800,4,5,1**	358.52	0.5520	0.5904	0.5318	0.5487
840,4,5,1	358.16	0.5428	0.5616	0.5419	0.5406
860,4,5,1	358.16	0.5428	0.5485	0.5419	0.5406
900,4,5,1	358.16	0.5428	0.5241	0.5419	0.5406

* PageView and Query Servers, DNS, MPC servers, UPLD servers

** Experiment time extended to 11,000～111,000ms

Table 5. Allocation of PageView Servers and Query Servers

Total Web Servers	PageView Servers	Query Servers	Ave. Response Time(ms)	Notes
900	724	176	357.92	4.12 : 1
900	810	90	490.65	9 : 1
800	644	156	357.92	4.12 : 1
600	483	117	358.13	4.12 : 1
500	402	98	365.96	2 times longer exp. time
500	404	96	371.14	same above
500	406	94	375.02	same above
500	400	100	363.74	same above
500	398	102	366.38	same above

로 증가시켜도 전반적인 시스템의 자원 활용률에 큰 변화가 나타나지 않았다. 따라서, 이 실험에서는 모든 웹서버가 0.55 이하의 활용률을 유지하려면 웹서버, DNS, MPC, UPLD 서버를 각각 (860, 4, 5, 1)로 배치해야 함을 알 수 있다.

마지막 제4단계에서는 제한된 수의 웹서버를 페이지 뷰와 지식 검색 전용으로 운영할 때 각각 몇 대씩 배치해야 평균 응답시간이 최소화되는가를 알아보기 위한 실험을 시행하였다. 이 경우 이전 실험의 결과를 반영하여, DNS는 4대, MPC 서버는 5대, UPLD 서버는 1대를 고정 배치하는 것으로 가정하였다.

이 문제를 Stochastic Optimization 형태의 문제로 기술하면 다음과 같다.

Minimize

$$Z = \frac{\lambda_1}{\lambda_1+\lambda_2} W_1 + \frac{\lambda_2}{\lambda_1+\lambda_2} W_2 \quad (1)$$

subject to

$$c_1 + c_2 \le c$$

$$0 < \frac{\lambda_1}{c_1\mu_1} < 1, \quad 0 < \frac{\lambda_2}{c_2\mu_2} < 1 \quad (2)$$

식 (1)에서 W_1, W_2는 각각 페이지 뷰 및 지식 검색을 나타내는 사용자 요구 유형별 평균 시스템 응답시간이고, λ_1, λ_2는 사용자 요구 유형 1, 2에 대한 도착율이며 c_1, c_2는 각 유형별로 할당된 서버의 수를 의미한다. Z는 $\lambda_1, \lambda_2, W_1, W_2$의 함수이고 W_1, W_2는 $\lambda_1, \lambda_2, \mu_1, \mu_2$(서비스 율)의 함수로서 Stochastic 속성을 갖고 있다. 식 (2)는 제약 조건으로 통상적인 큐잉 모델에서 설정하는 식이다. 위의 최적화 문제는 목적 함수의 형태를 알 수 없고 확률적 속성 때문에 통상적인 해석적 방식에 의한 최적화 방법으로는 해를 구할 수 없어서 이를 컴퓨터 시뮬레이션을 이용하여 분석해 보고자 한다.

본 실험에서는 우선, 지난 단계의 실험(Table 4)에서 나타난 바와 같이 비교적 충분한 수의 웹서버를 배치한 것으로 볼 수 있는 900대의 웹서버를 2가지 방법으로 배치하였다. 배치 방법으로는 첫째, 교통밀집도가 같게 되는 방식을 선택하여, 페이지 뷰와 지식 검색 서비스의 교통밀집도(ρ_1, ρ_2)를 같게 하는 c_1, c_2를 $\lambda_1/c_1\mu_1 = \lambda_2/c_2\mu_2$가 되도록 정하는 것이고, 둘째 각 서비스별 도착율에 비례하여 배치하는 것이다. 첫째 방식에 의하면 페이지 뷰 서버와 지식 검색 서버의 비율이 4.12대 1이고 둘째 방식에 의하면 9대 1의 비율이 설정된다. Table 5는 두 경우에 대한 평균 응답시간, 서버 활용률 등을 나타낸 것이다. 예상한 바와 같이 교통밀집도를 같게 하는 방식의 서버 배치가 월등한 성능을 보여주고 있다.

웹서버 수를 900대에서 지속적으로 감소시켜 600대까지 웹서버를 배치하더라도 평균 응답시간 측면에서 시스템 성능에 큰 차이를 보이지 않고 있다. 그러나 웹서버 수를 500대로 감소시키면 평균 응답시간이 증가하고 웹서버 활용률도 90%를 상회하는 것으로 나타났다. 이에 따라, 웹서버 수가 적을 때 평균 응답시간이 최소가 되는 웹서버 배치를 고려한 실험을 시행하였다.

웹서버 수가 500개일 때 페이지 뷰와 지식 검색 서비스에 배치되는 서버의 수에 따라 평균 응답시간의 변화가 가장 클 것이라는 예상아래 세밀한 웹서버 배치 실험을 한 결과 정확하게 교통밀집도가 같게 하는 배치에서 약간 벗어난 배치에서 최적의 성능(평균 응답 시간 기준)을 보여주고 있다. 그러나 교통밀집도를 같게 하는 배치에서

크게 벗어나지는 않는 것으로 나타났다. 향후 해석적 방법에 의한 분석이 이루어지면 정확한 최적 배치 방안이 제시될 수 있을 것으로 판단된다.

4. 결 론

본 논문은 웹서버 클러스터 시스템을 대상으로 효율적인 시스템 운용 방안에 대한 연구 결과이다. 본 논문에는 웹서버 클러스터 시스템에 대한 이전 연구를 기반으로 사용자 요구를 일정한 수준에서 유지시키면서 웹서버 클러스터 시스템을 구성하는 주요 요소인 DNS와 웹서버의 활용률을 일정한 수준 이하로 유지시키기 위해 얼마나 많은 웹서버를 어떻게 배치해야 하는가 살펴보았다. 또한, 제한된 수의 웹서버를 사용자 요구 유형에 따라 개별적으로 배치하는 경우 최적의 성능을 발휘하기 위한 방안을 모색하였다. 실험 결과 공동 사용 시스템이 전용 사용 시스템보다 응답시간과 자원 활용률 측면에서 우수한 성능을 보이는 것으로 나타났다. 그러나 전용 사용 시스템에서는 서비스 시간이 다소 단축될 수 있다고 판단되므로 반드시 공동 사용시스템이 더 우수한 성능을 나타낸다고 결론내릴 수는 없을 것으로 보인다. 마지막으로 전용 사용 시스템에서 서비스별 웹서버 배치 대수를 결정하는 문제는 교통밀집도를 같게 하는 서버 배치 방식이 최적으로 나타나지는 않았지만 상당히 우수한 배치 방식인 것으로 나타났다. 향후, 수학적 분석이나 세밀한 컴퓨터 시뮬레이션 실험을 통해 최적 서버 배치 방안을 결정하는 것도 좋은 연구 과제로 보인다. 대규모 웹서버 시스템에서 몇 대 정도의 웹서버 배치 이동이 전체 시스템 성능에 큰 변화를 가져오지는 않는 것으로 나타나, 이 문제는 대규모 웹서버 시스템의 경우보다 작은 규모의 웹서버 시스템이나 작업부하가 매우 높은 웹서버 클러스터 시스템에서 더 유용한 연구인 것으로 판단된다.

Reference

1. Trevor Schroeder, et. al., Scalable Web Server Clustering Technologies, IEEE Network, May/June, pp. 38-45, 2000.
2. Eunmi Choi, Performance test and analysis for an adaptive load balancing mechanism on distributed server cluster systems, Future Generation Computer Systems, 20, pp. 237-247, 2004.
3. Kim SC, Rhee Y, System Infrastructure of Efficient Web Cluster System to Decrease the Response Time using the Load Distribution Algorithm, J. of KIISE, Computing Practices, Vol. 10, No. 6, pp. 506-513, in Korean, 2004.
4. Kim JY, et. al., Effective Prioritized HRW Mapping in Heterogeneous Web Server Cluster, J. of KIISE, Computer Systems and Theory, Vol. 30, No. 12, pp. 708-713, in Korean, 2005.
5. Giovanni Pacifici, et. al., Performance Management for Cluster-Based Web Services, IEEE J. on Selected Areas in Communications, Vol. 23, No. 12, pp. 2333-2343, 2005.
6. Chung JY et. al., Efficient Content-based Load distribution for Web Server Clusters, J. of KIISE, Information Communication, Vol. 32, No. 1, pp. 60-67, in Korean, 2005.
7. Tzung-shi Chen, Kuo-Lian Chen, Balancing Workload based on Content Types for Scalable Web Server Clusters, Proc. of the 18th Int. Conf. on Advanced Information Networking and Application, Vol. 2, pp. 321-325, 2004.
8. Kim SC, Rhee Y, An Analysis and Comparison on Efficiency of Load Distribution Algorithm in a Clusterd System, J. of KIISE, Computing Practices, Vol. 12, No. 2, pp. 111-118, in Korean, 2006.
9. XiaoYi Lu, et. al., Request Distribution for Fairness with a Non-Periodic Load-Update Mechanism for Cyber Foraging Dynamic Applications in Web Server Cluster, The KIPS Transactions: Part-A, Vol. 14-A, No. 1, pp. 63-72, 2007.
10. Kaushik Dutta, et. al., ReDAL: An Efficient and Practical Request Distribution Technique for Application Server Clusters, IEEE Trans. on Parallel and Distributed Systems, Vol. 18, No. 11, pp. 1516-1528, 2007.
11. Zhongju Zhang, Weiguo Fan, Web Server load balancing: A queueing analysis, European J. of Operational Research, 186, pp. 681-693, 2008.
12. Jang HC, et. al., A Methodology for Performance Modeling and Prediction of Large-Scale Cluster Servers, J. of KIISE, Computing Practices and Letters, Vol. 16, No. 11, pp. 1041-1045, 2010.
13. Kang BJ, RYU HJ, Google vs. Naver, The Elctrocis Times, Book in Korean, 2008.
14. Chosun Ilbo, the good and bad sides of Internet Empire, NHN, nespaper in Korean, 2007.11.16., 2007.11.27.
15. Park JW, Analysis on the Performance Elements of Web Server Cluster Systems, J. of the Korea Society for Simulation, Vol. 19, No. 3, pp. 91-98, in Korean, 2010.
16. David Olshefski, Jason Nieh, Understanding the Management of Client Perceived Response Time, SIGMetrics/Performance '06, Saint Malo, France (2006).

17. Yongwha Chung, Daesung Moon, Taehae Kim, Jin-Won Park, Workload Dispatch Planning for Real Time Fingerprint Authentication on a Sensor-Client-Server Model," PDCAT 2004, LNCS 3320, pp. 833~838 (2004).
18. Wayne D. Smith, TPC-W: Benchmarking An Ecommerce Solution, www.tpc.org/tpcw
19. Huican Zhu, et. al., Demand-driven Service Differentiation in Cluster-based Network Servers, Proceedings of IEEE INFOCOM, Anchorage, U.S.A. (2001).
20. Valeria Cardellini, et. al., Enhancing a Web-Server Cluster with Quality of Service Mechanism, Proceedings of the IEEE Int'l Performance, Computing, and Communications Conference, Phoenix, U.S.A. (2002).

박 진 원 (jinon@hongik.ac.kr)

1971 서울대학교 공과대학 산업공학과 학사
1987 미국 오하이오 주립대학교 산업시스템공학과 석사, 박사
1988~1999 한국전자통신연구원 책임연구원
2000～현재 홍익대학교 게임소프트웨어전공 교수

관심분야 : 모델링&시뮬레이션, 시뮬레이션을 이용한 시스템 최적화, 공학교육