이기종 웹 클러스터 시스템에 대한 부하분산 알고리즘의 연구

이 영†

요 약

본 연구에서 이기종으로 구성된 웹 클러스터 시스템의 부하분산 알고리즘을 개발하고자 한다. 다수의 알고리즘을 제안하고, 동시사용자수에 의거하여 응답시간을 측정하고자 한다. 동적 가중치에 의한 부하분산 알고리즘과 고정가중치에 의한 부하분산 알고리즘을 비교하고 동적 가중치 알고리즘이 우수함을 입증하고자 한다. 또한 클러스터 시스템의 효율은 사용자수가 증가함에 따라 향상됨을 보이고자 한다.

A study of the load distributing algorithm on the heterogeneously clustered web system

Young Rhee†

ABSTRACT

In this paper, we develope algorithms that distribute the load on the heterogeneously clustered web system, The response time based on the concurrent user is examined for the suggested algorithms. Simulation experience shows that the response time using the dynamically weighted methods seems to have a good results compare to that with the fixed weighted methods. And, also the effectiveness of clustered system becomes better as long as the number of concurrent user increases.

키워드 : 웹 클러스터 시스템(Web Cluster System), 동시사용자(Concurrent User), 동적 가중치 방법(Dynamically Weighted Method), 고정 가중치 방법(Fixed Weighted Method)

1. 서 론

컴퓨터 시스템의 확장성에 관한 연구는 성능이 우수한 시스템을 개발하는 문제와 시스템에 부과되는 부하를 분산시키는 클러스터링 방법론으로 크게 구분된다. 이러한 접근은 효율성과 비용대비의 많은 응용사례와 연구결과를 보여주고 있다[2, 3]. Pfister[3]의 정의에 의하면 클러스터 시스템은 "전체 컴퓨터들이 상호 연결된 컴퓨터 집합으로 구성되며 이렇게 통합된 자원들이 하나의 단일 컴퓨터처럼 사용되는 병렬 혹은 분산 시스템의 한 종류를 의미한다"고 한다. 초기의 클러스터 방법론은 대부분 고속 계산을 위한 클러스터 시스템, HPC(High Performance Cluster)로 고도의 계산 능력이 요구되는 문제를 다수의 하부시스템(subsystem 이하 노드로 표기)에 분배하여 문제를 해결하는 방식이 주류를 이루었다. 그러나 인터넷의 발달과 함께 웹 사용의 증가 및 그 응용 범위의 확대로 인하여 기존의 단일 웹 서버 체제의 한계점에 이르게 되었다. 특히 웹 상에서의 전자상거래에 대한 괄목할 만한 성장 속에 웹 클러스터 시스템은 폭주하는 사용자의 요청에 대한 QoS 보장 측면에서 주목할 만한 시스템으로 부각되었다. 웹 클러스터 시스템은 부하분산 서버와 실제 웹 서비스를 수행하는 다수의 노드로 구성되어 사용자에게는 단일 웹 서버처럼 보이며 폭주하는 사용자의 요청을 분산시켜 QoS를 보장한다. 기존에 수행된 연구 중에서도 폭주하는 웹 요청을 분산시키기 위한 노력이 제시되었으며[4-6], 이러한 노력들은 대부분 하드웨어적 확장과 소프트웨어적인 확장으로 분류할 수 있다. 특히 소프트웨어적인 확장의 경우, 운영체계에 의존적인 연구가 대부분이며 이 연구 결과들은 동일한 사양의 노드들로 구성된 클러스터 시스템에 국한된 연구였다. 이러한 분야 중 고성능의 계산 능력에 초점을 맞춘 HPC 시스템이 대표적이다. HPC 시스템은 계산해야할 문제를 노드 수만큼 나누어 각 노드로 할당한 후 그 결과를 모두 통합하여 최종결과를 보여주기 때문에 동일한 사양의 시스템으로 구성

※ 2001년도 계명대학교 비사연구논문.
† 정 회 원 : 계명대학교 산업공학과 교수
논문접수 : 2002년 12월 3일, 심사완료 : 2003년 6월 16일

된 클러스터 시스템이 바람직하다. 그러나 웹 클러스터 시스템의 경우, 독립적인 요청이 각 노드로 할당되는 것이기 때문에 동일기종으로 구성된 시스템뿐만 아니라 성능이 다른 이 기종으로 구성된 웹 클러스터 시스템이 가능할 수 있다. 웹 클러스터 시스템의 부하분산에 가장 핵심적인 기능을 담당하는 것으로 부하분산 알고리즘이 있고, 초기의 부하분산 알고리즘들은 접속 수에 근거하여 개발되었다[4].

라운드로빈(Round-Robin) 알고리즘은 각 노드별로 부하를 동일하게 분배하는 방식으로 네트워크의 연결 수에 기반을 두고 있어서 라운드로빈 DNS 보다 효과적이라 할 수 있다. 가중치 기반 라운드로빈(Weighted Round-Robin) 알고리즘은 노드 마다 서로 다른 처리능력 고려하여 서버마다 부하의 양을 다르게 분산시키지만 부하가 폭주하는 경우, 부하의 크기를 고려하지 않고 각 요청을 동일하게 간주하므로 단 시간에 폭주하는 시스템의 경우 부적합하다고 할 수 있다. 최소 접속 수(Least Connection) 알고리즘은 새로운 서비스 요청이 발생할 경우, 웹 클러스터 시스템을 구성하는 각각의 서버 중 현재 가장 적은 연결 수를 가진 서버에 할당하는 방식이다. 가중치 기반 최소 접속 수, WLC(Weighted Least Connection) 알고리즘은 웹 클러스터 시스템을 구성하는 노드 각각의 성능에 따라 가중치를 부여한 후, 가중치가 높은 서버에게 비율에 근거하여 연결을 할당하는 방식이다. 부하분산 과정은 W_i를 n개의 서버 각각에 대한 가중치, $C_i(i = 1, 2, \cdots, n)$를 n개의 서버 각각이 현재 가지고 있는 접속 수라하고 S를 웹 클러스터 시스템의 전체 접속 수라 할 때, 새로운 서비스 요청이 들어올 경우, n개의 서버 중 접속수와 가중치의 비율이 최소인 임의의 서버 j에게 접속을 할당하게된다. WLC 알고리즘은 초기에 가중치를 고정하기 때문에 단순히 접속수에 의거한 방식을 취하고 있다고 할 수 있으며 노드들의 상태정보, 즉 노드들의 부하정도를 고려하지 않고 있다.

이상과 같이 기존의 부하분산을 위하여 사용된 알고리즘들은 주로 동일기종으로 구성된 웹 클러스터 시스템에 한정된 경우가 대부분임을 알 수 있고, 이 알고리즘들은 부하분산 과정에서 동일 사양의 웹 클러스터 시스템일 경우에는 대체적으로 우수한 알고리즘으로 인식되어 왔다. 그러나 현실적으로 동일 기종이 아닌 서로 다른 이 기종으로 웹 클러스터 시스템을 구성해야 한다면 이제까지 제시된 부하분산 알고리즘은 해결책이 되지 못할 수 있다는 점을 인지할 필요가 있다. 따라서 서로 다른 기종의 웹 클러스터 시스템의 부하분산을 위하여 시스템의 상태정보가 수집되어, 이에 상응하도록 적절히 접속 요청을 분배함이 바람직하다고 판단된다. 그러므로 본 연구에서는 성능과 기타 물리적인 환경이 서로 다른 이 기종 웹 클러스터 시스템에서 빈번한 사용자의 접속 요청을 적절하게 할당하는 방법론에 대한 몇 가지 대안을 제시하고자 한다.

본 연구의 2절에서는 시스템의 상태정보를 이용한 동일 기종 웹 클러스터 시스템에 대한 부하분산 알고리즘을 소개한다. 또한 간단한 실험을 통하여 동시 사용자수의 변화에 따른 기존 알고리즘과 제안된 부하분산 알고리즘 적용 하에서의 응답시간을 검토하고자 한다. 3절에서는 이 기종 웹 클러스터 시스템의 실험환경과 부하분산 방법론을 제시하고자 한다. 4절에서는 제시된 방법론에 의한 모의실험을 이용하여 수집된 자료를 분석하여 부하분산 방법의 지표를 제시한다. 끝으로 5절에서는 결론을 제시한다.

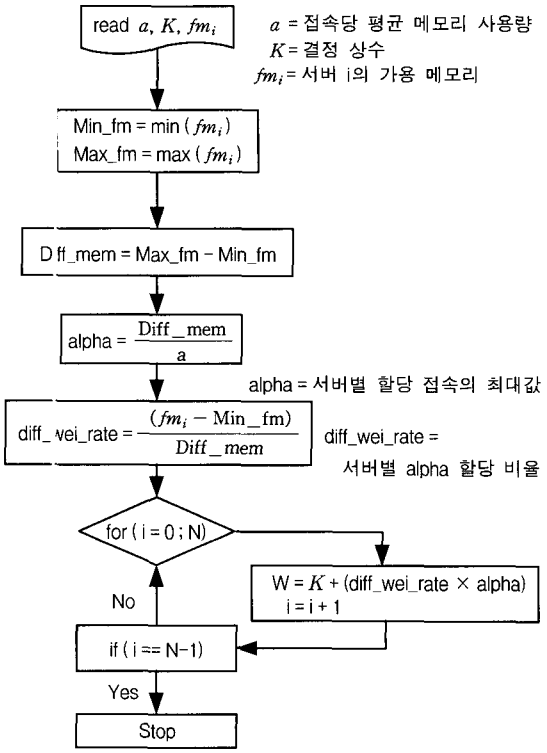
2. 동일기종 웹 클러스터 시스템

본 절에서는 동일기종으로 구성된 웹 클러스터 시스템의 부하정도를 나타내는 상태 정보에 근거한 부하분산 알고리즘을 제안하며, 기존의 알고리즘과 간략히 비교하고자 한다.

2.1 부하분산 알고리즘

본 연구의 실험 환경은 1대의 부하분산 서버와 성능이 동일한 다수의 노드를 가지는 동일 기종 웹 클러스터 시스템과 웹 클러스터 시스템에 부하를 부과할 부하생성 서버로 구성되었다. 웹 클러스터 시스템의 각 노드의 사양은 동일 사양으로 CPU와 메모리가 133MHz/32Mbyte로 구성하였다. 사용자들의 요청 처리는 동일 회선에서 받아들이고, 웹 클러스터 시스템의 내부 네트워크를 통해 각 노드에 요청이 할당된다. 각 노드는 할당받은 요청을 처리한 후 각 노드가 가지는 별도의 회선을 통해 직접 사용자에게 결과를 보내게 된다. 노드의 부하 측정을 위해 각 노드의 상태정보, 즉 각 노드가 보유하고 있는 가용한 메모리(free memory) 양과 버퍼 메모리를 기초로 부하의 정도를 판단하였다. 이 알고리즘에서 중요한 결정 변수로 a, alpha 그리고 K가 있다. 이중 K는 웹 클러스터 시스템 내에 있는 특정 서버로 1분 동안 접속하는 평균 접속수를 의미하며, a는 하나의 접속당 평균적으로 소모되는 메모리의 양을 의미하고, alpha는 최대 추가 접속수를 의미한다. alpha는 각 서버의 상태정보 중에서 최대 가용메모리와 최소 가용메모리의 차이를 평균 접속 당 할당 메모리 a로 나눈 값으로, 특정 서버에 할당될 추가 접속수는 그 서버의 가용메모리와 최소 가용메모리의 차이, 그리고 최대 가용메모리와 최소 가용메모리의 차이를 선형비율로 산출한 값이다. 따라서 추가 접속수는 0과 alpha의 사이 값이라고 할 수 있다. 가중치 선정 과정에서 K는 각 서버로 기본적으로 접속된 분당 평균 접속수를 의미한다. K값의 산정은 부하분산의 주요 요소로서 K값이 작은 경우는 부하분산이 alpha값에 의존적이고, K값이 큰 경우는 상대적으로 alpha값에 덜 의존적이다. 이 알고리즘의 구현으로 1분 간격으로 서버의 상태정보를 검토하여 최대 alpha 개의 접속을 더 할당하고자 하는 것으로 K의 선택은 부하분산에 중요한 요소이므로 분당 접속회수(turn around)

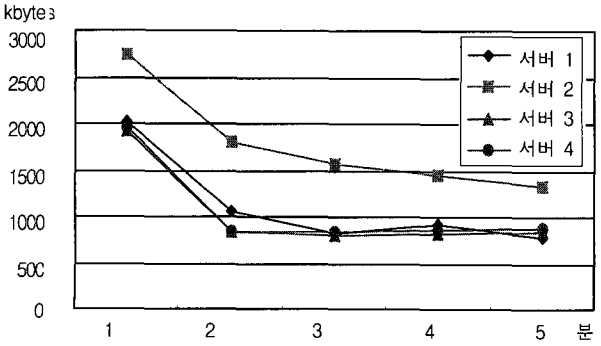
와 동시사용자를 곱한 값을 이용하여 산정 하였다. (그림 2.1)은 동적 부하분산 알고리즘의 흐름도를 나타낸 것이다.



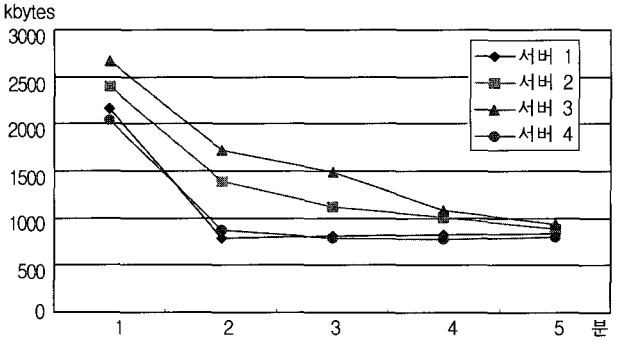
(그림 2.1) 동일기종 시스템의 동적 부하분산 알고리즘

2.2 실험 결과 및 분석

동일 기종 웹 클러스터 시스템에 대한 실험은 동시사용자수를 증가시키면서, 한편으로는 기존 알고리즘(WLC)과 제안된 알고리즘을 사용한 동일 기종 웹 클러스터 시스템들의 가용메모리의 사용 정도와 응답시간을 측정하는 실험을 수회 반복하였다. 실험 결과는 (그림 2.2)와 (그림 2.3)에서 보는 바와 같이 기존 알고리즘과 제안된 알고리즘에 대한 가용메모리의 사용 정도에서, 기존 알고리즘이 가용메모리를 적절히 사용하지 못하는 반면, 제안된 알고리즘을 적용한 경우에는 전체 노드의 메모리를 공평하게 사용하려는 경향을 볼 수 있었다.



(그림 2.2) 기존 알고리즘의 가용 메모리의 변화



(그림 2.3) 제안된 알고리즘의 가용 메모리의 변화

또한 평균 응답시간에 대한 실험에서도 <표 2.1>과 <표 2.2>의 결과에서처럼 제안된 알고리즘이 기존 알고리즘 보다 대체적으로 응답시간이 더 단축되어 노드의 상태정보를 반영한 알고리즘을 적용함으로써 동일기종 웹 클러스터 시스템의 성능을 향상시킬 수 있다는 것을 알 수 있었다.

〈표 2.1〉 서버 수가 2대인 경우의 응답시간의 비교

실험군 \ 동시사용자	30	50	70	90
기존 알고리즘	0.017	0.040	0.104	0.167
제안된 알고리즘	0.017	0.030	0.091	0.143

〈표 2.2〉 서버의 수가 4대인 경우의 응답시간의 비교

실험군 \ 동시사용자	30	50	70	90
기존 알고리즘	0.014	0.016	0.018	0.020
제안된 알고리즘	0.013	0.014	0.016	0.018

3. 이 기종 웹 클러스터 시스템

3.1 실험 환경 및 방법

본 연구를 위해 구성한 웹 클러스터 시스템은 동일기종 웹 클러스터 시스템에 대한 실험 환경과 동일하며 차이점은 웹 클러스터 시스템을 구성하고 있는 각 노드의 성능이 서로 다른 이기종으로 구성되었다는 것이다. 참고적으로 본 연구에서 사용된 서버의 사양은 CPU와 메모리 기준으로 800MHz/256Mbyte, 500MHz/128Mbyte, 166MHz/64Mbyte, 133MHz/32Mbyte이며, 노드의 수가 2대로 구성된 웹 클러스터 시스템의 구성은 고 사양의 시스템을 선택하였다. 실험의 진행은 웹 서비스를 요청하도록 표본으로 프로그램을 작성하였고, 프로그램의 실행은 웹 부하생성 서버에서 동시사용자수를 200, 300, 400, 500으로 증가하면서 웹 클러스터 시스템에 부과실험을 20분간 지속한다. 데이터는 1분 주기로 서버의 가용메모리의 양과 가중치를 수집하였고 마지막 20분 시점에서 평균 응답시간을 측정하였다. 그리고 실험은 웹 클러스터 시스템의 실제 서버 수와 부하크기에 따라 각 40회 씩 반복하여 수행하였으며, 5회 실험 반복 후 시스템

을 재부팅하여 매 실험의 이전 부하에 의한 영향을 제거하였다. 그리고 웹 클러스터 시스템의 실제서버의 수는 2대와 4대로 구성된 시스템에 대하여 각각 실험을 수행하였다. 실험 대상 시스템으로는 노드의 부하를 고려하지 않은 웹 클러스터 시스템(가중치가 모두 동일하게 고정된 웹 클러스터 시스템)과 서버의 부하를 고려하여 가중치를 변화시키는 제안된 몇 개의 알고리즘을 적용한 웹 클러스터 시스템에 대하여 응답시간 및 가용메모리의 사용 정도를 측정하였다.

3.2 동적 부하분산 알고리즘

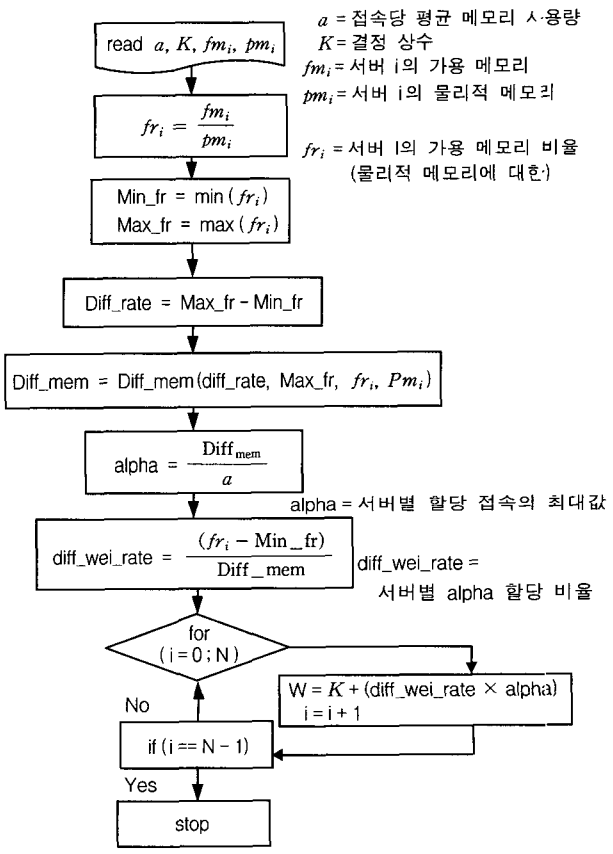
3.2.1 노드의 성능 지표 및 차이와 관련한 문제점

일반적인 웹 서버의 경우, 요청을 처리하기 위해 CPU와 메모리 등 시스템의 자원을 사용하게 된다. 그러나 웹 서비스의 성능의 척도는 CPU의 속도보다는 데이터의 버퍼링 기능을 담당하는 메모리에 큰 영향을 받는다고 한다[1]. 실험 결과에 의하면 부하 크기에 관계없이 CPU의 유휴시간 비율이 평균 75~90%로 일정하게 나타났다. 이 결과를 해석하면, 웹서버에 있어서 CPU가 중요 성능 지표가 될 수 없다고 할 수 있다. 물론 CGI와 같이 CPU 의존도가 높은 웹 어플리케이션에서 부하가 문제인 경우에는 논외의 대상이 된다. 그러므로 웹 서버의 중요 성능 지표로 물리적 메모리 양과 버퍼메모리(이하 물리적 메모리로 표기)가 우선한다고 할 수 있다. 그러나 메모리의 경우, 각 노드의 물리적 메모리의 양이 모두 다른 이 기종 웹 클러스터 시스템이 요청 처리를 하기 위해 각 노드는 일정한 양의 메모리를 계속하여 소비하게되며 가용한 메모리가 더 이상 없을 경우 노드들은 메모리 스와핑을 하게되어 시스템의 성능은 극도로 나빠질 수 있다. 그리고 각 노드가 보유한 물리적 메모리 양이 선형으로 증가한다고 하여 각 노드의 성능 차이 또한 선형이라고 할 수 없고, 요청이 폭주하게 되는 경우, 가장 적은 양의 메모리를 가진 노드부터 그 성능이 급격하게 떨어지게 되어 전체 웹 클러스터 시스템의 성능 또한 저하하게 된다. 이러한 이유로 각 노드의 가중치가 같다고 전제하거나, 고정시키는 정책은 웹 클러스터 시스템의 성능 저하를 가속시키는 원인이 될 수 있다. 그러므로 노드의 상태정보를 기초로 하여 각 노드의 가중치를 시기 적절하게 변경시켜 줄 필요가 있다.

3.2.2 부하분산 알고리즘

본 연구에서 실험 대상으로 제안된 알고리즘은 크게 두 부류로 구분할 수 있다. 첫째는 각 노드의 가중치를 고정하는 고정 가중치 방법과, 시스템의 상태정보에 의거 일정 시간 간격으로 가중치를 변경하는 동적 가중치 방법이라고 할 수 있다. 고정 가중치 방법에는 기존 알고리즘, WLC에 근거하여 각 노드의 가중치를 모두 1로 고정하는 경우와 본 연구에서 제안하는 각 노드의 가중치를 보유한 물리적 메모리 양에 비례하여 가중치를 고정하는 경우로 세분화된다. 동적 가중치 방법으로는 일정 시간 간격으로 시스템의 상태정보, 즉 가용메모리의 차이에 의한 가중치 부여방법이다. 또 다른 동적 가중치 방법으로는 각 노드의 가용메모리와 물리적 메모리의 비율에 의하여 가중치를 부여하는 방법이다. 이상에서 제시된 알고리즘을 자세히 소개하면 다음과 같다.

가중치를 모두 1로 고정하는 경우는 각 노드의 물리적 메모리 차가 성능에 영향을 주지 않는다고 가정하는 경우이다. 그리고 본 연구에서 제안하는 대안 1은 초기 물리적 메모리의 비율에 근거하여 고정된 가중치를 부여하는 경우로서, 예를 들어 32Mbyte와 64Mbyte의 두 기종으로 이루어진 시스템에서 초기에 가중치를 1 : 2로 부과하는 방법이다. 따라서 가중치는 시스템 상태정보와 관계없이 고정된 값이 부여된다. 대안 2의 알고리즘은 시스템 운영 중에 조사된 가용메모리의 양을 근거로 하여 차등화 된 가중치를 부여하는 방법으로 2절에서 소개된 동일기종 웹 클러스터 시스템의 부하분산 알고리즘과 정확히 일치한다고 보면 된다. 마지막으로 대안 3의 알고리즘은 대안 2의 알고리즘에서처럼 시스템의 상태정보를 이용하여 시스템의 부하 정도를 판단하고 기타 알고리즘의 흐름은 대안 2와 유사하다. 그러나 부하분산의 판단 근거는 시스템 운영 중에 조사된 가용메모리와 초기 물리적 메모리의 비율에 따라 가중치를 부여하는 방법



(그림 3.1) 가용메모리의 비율에 근거한 알고리즘

사용한다. 예를 들어 서버 1의 초기 물리적 메모리는 32Mbyte, 서버 2의 초기 물리적 메모리는 64Mbyte 이었고, 운영 중에 서버 1과 서버 2의 가용메모리가 16Mbyte, 48Mbyte라면 서버 1의 사용 가능 비율은 $\frac{16}{32}$, 즉 0.5이고, 서버 2의 사용가능 비율은 $\frac{48}{64}$, 즉 0.75로서 서버 2에 전체 부하의 25%를 더 부과시키는 알고리즘이다. (그림 3.1)은 대안 3 알고리즘에 대한 흐름도를 나타낸 것으로, 여기에서 Diff_mem은 가용메모리의 비율이 가장 높은 서버에 대해 전체 물리적 메모리에 비율의 차를 곱한 것으로, 접속 당 메모리의 소모량을 의미한다.

4. 실험 결과 및 분석

기존 알고리즘과 대안 1, 대안 2, 대안 3의 알고리즘을 적용한 이기종 웹 클러스터 시스템에 대한 실험을 수행하여 노드별 가용메모리의 사용 경향과 스와핑 발생회수, 그리고 응답시간에 대한 실험 결과를 분석하고자 한다.

노드별 가용메모리의 사용 경향에 대한 실험 결과에서 가중치가 1로 고정된 기존 알고리즘의 경우, 웹 클러스터 시스템을 구성하는 각 노드 사이의 가용메모리 차이에 관계없이 노드별로 독립적으로 가용메모리가 사용된다. 즉 가용메모리가 대부분 소진되어 극심한 성능 저하가 발생하는 노드가 발생하더라도 가용메모리가 많이 남아있는 노드로 부하가 분산되지 않는다. 그러나 대안 1, 대안 2, 대안 3의 경우 기본적으로 각 노드별 성능 차이를 인정하는 알고리즘이기 때문에 가용메모리가 소진되는 노드가 발생할 경우 가용메모리가 남은 노드로 부하를 분산시키고, 단시간에 특정 노드의 가용메모리가 소진되는 현상이 발생하지는 않는다. 특히 대안 2, 대안 3의 경우, 노드별 성능 차이를 인정하고, 부하 정도에 따라 다시 노드의 가중치가 변경되므로 가용메모리가 소진된 노드가 발생할 경우, 상대적으로 가용메모리가 많이 남은 노드로 부하를 분산시키려는 경향이 목격되었다. 이것으로 기존의 알고리즘보다 제안된 알고리즘, 특히 노드별 상태정보에 근거하여 부하를 분산시키는 대안 2와 대안 3이 보다 좋은 결과를 보임을 알 수 있었다. 이러한 결과는 적용된 알고리즘에 따른 분당 평균 스와핑 발생 횟수에 대한 결과에서도 알 수 있다.

<표 4.1>에서 보는 바와 같이 기존 알고리즘을 적용한 경우, 물리적 메모리가 많은 노드일수록 swap 발생 횟수가 현저히 낮아져 노드 3에서부터는 swap이 전혀 발생하지 않았으나, 물리적 메모리가 가장 낮은 노드 1은 분당 swap-in이 평균 703회, swap-out이 평균 256회로 실험 시간 동안 지속적으로 발생하였고 이것이 웹 클러스터 시스템의 응답시간에 영향을 미치는 것을 알 수 있었다. 이와 달리 대안 1, 대안 2, 대안 3의 알고리즘을 적용한 경우, 노드 1의 swap 발생 회수가 기존 알고리즘을 사용한 경우보다 현저해졌음을 알 수 있었고, 상대적으로 물리적 메모리가 많은 노드 2, 노드 3, 노드 4에서는 swap이 전혀 발생하지 않아 응답시간을 개선할 수 있었다. 이러한 이유는 메모리의 사용 경향에서 밝힌 바와 같이 노드별 성능을 고려하지 않고 부하를 부과하는 기존 알고리즘의 경우, 가용메모리가 모두 소진되어 가상메모리를 사용하기 위하여 swap이 발생하기 때문이며, 이것은 응답시간에 큰 영향을 미치는 요소라고 할 수 있다.

〈표 4.1〉 적용 알고리즘 별 스와핑 발생 회수

노드 (물리적 메모리 Kbytes)	노드 1 (32)		노드 2 (64)		노드 3 (128)		노드 4 (256)	
적용 알고리즘	swap		swap		swap		swap	
	in	out	in	out	in	out	in	out
기존 알고리즘	703	256	간헐적 발생		0	0	0	0
대안 1	40	23	0	0	0	0	0	0
대안 2	20	13	0	0	0	0	0	0
대안 3	25	14	0	0	0	0	0	0

노드의 수에 따른 기존 알고리즘과 대안 알고리즘들의 응답시간에 대한 실험으로 <표 4.2>와 <표 4.3>의 이기종 웹 클러스터 시스템에 대한 평균 응답시간의 측정 실험 결과를 얻을 수 있었다. 여기에서 언급하는 응답시간은 단일 웹 페이지에 포함된 하나의 콘텐츠에 대한 응답시간을 평균한 것이다. <표 4.2>에서 보는바와 같이 이기종 웹 클러스터 시스템을 구성하는 노드의 수가 2대인 경우, 동시접속자수가 증가 할 수록 대안 2와 대안 3이 빠른 응답시간을 보인다. 동시접속자 수가 200에서 기존 알고리즘과 대안 3의 응답시간의 차이가 0.062초 개선된 반면, 동시접속자 수가 500으로 증가한 실험에서는 0.413초가 개선되어 동시접속자 수가 증가할수록 성능 개선 효과가 높아진다고 할 수 있다. 또한 대안 1보다 대안 2, 대안 3이 전체적으로 보다 나은 개선 효과를 보이는데 이것으로 가중치를 고정시키는 정책보다는 노드의 부하에 따라 동적으로 가중치를 변화시켜주는 것이 효과적이라는 것을 알 수 있었다. 이러한 이유는 동시접속자가 증가할수록 물리적 메모리가 적은 노드들이 가용메모리를 모두 소진하여 잦은 swap이 발생하고 이로 인해 전체 웹 클러스터 시스템의 성능을 저하시키기 때문이다. 노드의 수가 4대인 경우, 노드의 수가 2대인 웹 클러스터 시스템보다 4대인 경우가 개선효과가 측면에서 볼 때 개선효과의 폭이 둔화된 것을 볼 수 있는데 이것은 실험에서 부과한 부하의 크기가 가장 좋은 성능을 보이는 노드가 충분히 소화할 수 있는 정도의 부하였기 때문에 그 변화 폭이 둔화된 것처럼 보인 것이라 판단된다. 그러나 전체적으로 기존 알고리즘보다 대안 1, 대안 2, 대안 3이 더 나은 개선효과를 보였음을 알 수 있다.

〈표 4.2〉 노드의 수가 2대인 경우의 응답시간

동시접속자수	적용 알고리즘	응 답 시 간
200	기존 알고리즘	0.418
	대안 1	0.378
	대안 2	0.367
	대안 3	0.356
300	기존 알고리즘	0.844
	대안 1	0.629
	대안 2	0.639
	대안 3	0.600
400	기존 알고리즘	1.022
	대안 1	0.888
	대안 2	0.877
	대안 3	0.876
500	기존 알고리즘	1.500
	대안 1	1.129
	대안 2	1.068
	대안 3	1.087

〈표 4.3〉 노드의 수가 4대인 경우의 응답시간

동시접속자수	적용 알고리즘	응 답 시 간
200	기존 알고리즘	0.251
	대안 1	0.250
	대안 2	0.249
	대안 3	0.247
300	기존 알고리즘	0.510
	대안 1	0.441
	대안 2	0.392
	대안 3	0.480
400	기존 알고리즘	0.735
	대안 1	0.574
	대안 2	0.542
	대안 3	0.594
500	기존 알고리즘	0.984
	대안 1	0.745
	대안 2	0.738
	대안 3	0.738

실험 결과를 종합하면, 고정 가중치보다는 동적 가중치 방법이 보다 나은 가용메모리의 사용경향을 보이며, swap 발생 회수를 감소시키고, 이로 인하여 보다 빠른 응답시간을 얻을 수 있었다. 또한 웹 클러스터를 구성하는 노드의 수에 무관하게 동시사용자 수가 적은 경우 대안 별 차이가 없었으나, 동시사용자의 수가 증가할 수록 대안 별 웹 클러스터 시스템의 효과는 더욱 개선됨을 볼 수 있었다.

5. 결 론

웹 클러스터 시스템은 독립적인 요청이 각 노드로 할당되기 때문에 동일기종으로 구성된 시스템뿐 만 아니라 성능이 다른 이 기종으로 구성된 웹 클러스터 시스템으로 구성할 수 있다. 이런 이유로 요청이 폭주하는 웹 서버의 확장을 고려하는 경우, 기존 시스템과 새로운 시스템을 통합하여 이기종 웹 클러스터 시스템을 구성하는 것은 의미가 있다고 할 수 있다. 또한 기존의 웹 클러스터 시스템에서 부하분산을 위해 사용한 알고리즘들은 주로 동일기종으로 구성된 웹 클러스터 시스템에 한정된 경우가 대부분이다. 그러나 현실적으로 동일 기종이 아닌 서로 다른 이 기종으로 웹 클러스터 시스템을 구성해야할 경우, 이제까지 제시된 부하분산 알고리즘은 전혀 해결책이 되지 못할 수 있기 때문이다. 이에 본 연구는 이기종 웹 클러스터 시스템의 부하를 고려하는 알고리즘을 바탕으로 응답시간을 향상시킬 수 있는 방법에 관하여 연구를 수행하였다. 이를 위해 동시사용자 수와 이기종 웹 클러스터 시스템의 노드 수와 기존 알고리즘 및 대안 알고리즘들에 대한 실험을 수행하여 고정 가중치보다는 동적 가중치 방법이 양호한 가용메모리 사용경향을 보이고, 대안 알고리즘을 사용함이 빠른 응답시간을 얻을 수 있고, 동시사용자의 수가 증가할수록 웹 클러스터 시스템의 효과가 더욱 개선됨을 알 수 있었다.

참 고 문 헌

[1] 권원상, 역. 웹 성능 최적화. Patrick Killelea, 한빛미디어, 2000.
[2] Daniel A, Menace, virgilio A. F. Almeda, "*Capacity planning for WEB PERFORMANCE Metrics, Models, & Methods*," Pretice Hall PTR, New Jersey, 1998.
[3] Gregory F. Pfister, "*In Serach of Clusters Second Edition*," Prentice-Hall PTR, New Jersey, 1998.
[4] Wensong Zhang, The paper "Linux Virtual Server for Scalable Network Services," Ottawa Linux Symposium, 2000.
[5] Cisco Systems, "Cisco LocalDirector 400 Series Content Switches Relevant Technologies," http://www.cisco.com/en/US/products/hw/contnetw/ps1894/prod_relevant_tech.html, 2002.
[6] microsoft corporation, "Server Cluster Architecture," http://www.microsoft.com/windows.netserver/techinfo/overview/servercluster.mspx, 2002.

이 영
e-mail : yrhee@kmu.ac.kr
1983년 고려대학교 산업공학과(공학사)
1885년 고려대학교 산업공학과(공학석사)
1990년 University of Oklahoma 산업공학과 (MS)
1994년 North Carolina State University OR/CS(Ph.D)
1995년 삼성 SDS 정보기술연구소 수석연구원
1998년~현재 계명대학교 산업공학과 조교수
관심분야 : Network 모델링과 Performance 분석, 웹 서버 capacity planning