웹 비즈니스의 고가용성을 위한 동적 다중 웹 분산 클러스터 그룹 모델

이 기 준† · 박 경 우†† · 정 채 영†††

요 약

인터넷의 급속한 성장과 더불어 이를 이용한 다양한 웹 비즈니스 환경은 가상공간을 기반으로 한 새로운 생활환경을 만들어 가고 있다. 그러나 이러한 인터넷 활용과 사용자의 급증은 과중한 전송량과 시스템의 부하문제를 야기 시켰으며 이를 해결하기 위한 여러 방안으로 병렬 클러스터 시스템이 모색되어지고 있다. 본 논문에서는 다수의 저가, 저속의 시스템을 이용한 MPP 동적 분산 서브 클러스터 그룹(MPP dynamic sub-cluster group)을 구성하고 구축된 서브 클러스터 그룹을 단일한 가상 IP에 묶어 클라이언트의 서비스 작업을 수행할 수 있는 다중 분산 클러스터 그룹(multi distributed cluster group)을 제안한다. 제안한 다중 분산 클러스터 그룹은 LVS를 기반으로 한 상층구조와 SC-Server를 중심으로 한 동적 서브 클러스터 그룹으로 구성되며 사용자로부터 요구된 작업량을 병렬 처리한다. 제안한 클러스터 그룹은 일반 Web Service 작업이외에 Database 관리, 병렬 연산을 요구하는 대규모 연산작업, 서비스 폭주에 따른 부하조절등에 효율적으로 이용될 수 있다.

Dynamic Multi-distributed Web Cluster Group Model for Availability of Web Business

Kee-Jun Lee† · Kyung-woo Park†† · Chai-Yeoung Jung†††

ABSTRACT

With the rapid growth of the Internet, various web-based businesses are creating a new environment in an imaginary space. However, this expanding Internet and user increase cause an overflow of transmission and numerous subordinate problems. To solve these problems, a parallel cluster system is produced using different methods. This thesis recommends a multi-distribution cluster group. It constructs a MPP dynamic distribution sub-cluster group using numerous low-priced and low-speed systems. This constructed sub-cluster group is then connected with a singular virtual IP to finally serve the needs of clients (users). This multi-distribution cluster group consists of an upper structure based on LVS and a dynamic serve cluster group centered around an SC-server. It conducts the workloads required from users in a parallel process. In addition to the web service, this multi-distribution cluster group can efficiently be utilized for the calculations which require database controls and a great number of parallel calculations as well as additional controls which result from the congestion of service.

키워드 : MPP, 분산 시스템, 클러스터링, 웹 비즈니스

1. 서 론

최근 인터넷(internet)의 비약적인 성장과 컴퓨터 시스템의 눈부신 발전은 기존의 시간적, 공간적 개념을 뛰어넘은 새로운 생활 문화공간을 만들어 가고 있다. 사용자는 인터넷을 이용하여 원하는 정보를 취득하며 자신의 자료 및 자원을 다른 이들과 공유, 활용할 수 있게 되었다. 따라서 서버에서 처리해야할 데이터의 양과 사용자가 요구하는 정보의 전송량은 급속히 증대되고 있으며, 과중한 데이터양은 네트워크의 병목현상(bottle neck), 데이터 처리량에 따른 시스템 부하(system load)등의 문제점들을 야기 시켰다. 현재 이러한 문제점들을 극복하고 다양한 웹 비즈니스 환경에서 효율적인 시스템 운영을 위한 여러 방안들이 모색되어지고 있으며 이중 저가의 중형서버(middle server)를 병렬 클러스터 시스템으로 구성한 대형 병렬 서버가 등장하게 되었다[1, 2].

고속의 네트워크에 연결된 서버들의 클러스터는 성능이 뛰어나고 가용성이 높은 시스템을 구축하는데 유효하며, 단일 프로세서 시스템(single processor system)이나 엄격한 다중 프로세서 시스템(multi processor system)에 비하여 확장성이 뛰어나고 적은 투자비용에 비하여 효율성과 신뢰성이 우수하다. 이러한 클러스터 시스템 구조에는 벡터 컴퓨터(vector computer)[3, 4], SMP 클러스터 모델[5, 6], MPP

† 정 회 원 : 조선대학교 대학원 전산통계학과
†† 정 회 원 : 광주보건대학 전산정보처리과 교수
††† 종신회원 : 조선대학교 수학 · 전산통계학부 교수
논문접수 : 2001년 4월 19일, 심사완료 : 2001년 7월 12일

클러스터 모델, P2P모델등이 있다. 벡터 컴퓨터는 많은 데이터를 같은 명령으로 처리할 경우 큰 효과를 기대할 수 있으나 고성능의 프로세서를 사용하기 때문에 가격이 비싸고 유지 관리가 힘들며, SMP 클러스터 모델은 연산노드가 하나 이상의 대칭적 마이크로 프로세서로 구성되어 지역적인 메모리를 공유하는 시스템으로 다수의 프로세서들이 상호 연결망을 통하여 기억장치들을 공유하므로 통신망에 연결된 자원들은 어느 프로세서든지 쉽게 사용할 수 있는 장점을 지녔지만 충분한 대역폭을 제공하여야 하기 때문에 대규모 시스템을 구축할 때 비용이 많이 들고 복잡해지며 확장성이 약하다는 단점을 지니고 있다[5, 6]. 이에 반하여 MPP 클러스터 구조는 13~38MB/Sec의 메시지 전송속도를 지원하는 다량의 노드들이 연결된 시스템으로 메모리가 물리적으로 각 노드에 분산되어 있지만 논리적으로 단일한 주소공간을 형성하므로 확장성과 프로그램 작성이 용이하다. 또한 MPP 구조는 확장성이 우수하여 수천 개까지 노드의 수를 증가시킬 수 있으며 메모리를 물리적으로 각 노드에 분산시켜서 다수의 프로세스가 동시에 메모리를 사용하였을 때 발생하는 병목현상을 제거할 수 있다.

본 논문에서는 MPP 클러스터 모델을 이용하여 다수의 저가(low price), 저속(low speed)의 시스템에 대한 작업량, 시스템의 성능분석, 네트워크 대역폭등을 이용한 동적 다중 서브 클러스터(dynamic multi sub-cluster)를 형성하고 구축된 서브 클러스터 그룹을 단일한 가상 IP에 묶어 클라이언트의 서비스 작업을 수행할 수 있는 MPP 동적 다중 분산 클러스터 그룹(MPP dynamic multi distributed cluster group)을 제안한다. 기존의 클러스터 방식은 중,소형 서버(server)를 기반으로 구축한 반면 제안한 클러스터 방식은 주위에서 쉽게 접할 수 있는 일반 시스템을 이용함으로 경제적인 효과를 극대화시킬 수 있다. 또한 개별적인 서브 클러스터 그룹은 다수의 시스템을 MPP 분산 병렬 컴퓨팅구조(distributed parallel computing structure)로 구축함으로서 처리의 효율을 극대화시킬 수 있다.

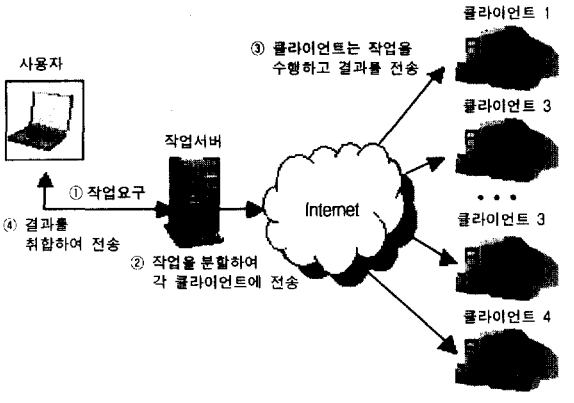
본 논문은 다음과 같이 구성되어 있다. 2장에서 클러스터 그룹을 구성하기 위한 기존의 구성방식에 관하여 살펴보고, 3장에서 제안한 서브 클러스터 그룹의 구성 및 구축방안에 대하여 알아본다. 그리고 4장 구현 및 실험을 통하여 제안한 방식의 경제적 효율성 및 처리효율의 고가용성을 확인하고 5장에서 결론을 맺는다.

2. 클러스터링 모델

2.1 P2P를 이용한 클러스터링

P2P방식은 기존의 클라이언트/서버 구조상에서 발생될 수 있는 서버의 트래픽 집중문제를 해결하기 위하여 클라이언트 상호간의 분산, 협력 방식으로 제안되었다. P2P는 이미 냅스터(napster), 그누텔라(gnutella)등을 통하여 이미 생활주변에서 쉽게 접할 수 있으며, 현재 이러한 P2P방식을 이용하여 네트워크에 연결된 유휴의 자원을 동적인 인터넷 어플리케이션으로 묶어 병렬구조를 가진 분산화된 가상의 슈퍼컴퓨터를 구성하기 위한 연구가 수행되고 있다.

P2P방식은 분산 컴퓨팅방식을 통하여 네크워크에 등록된 하드웨어 자원을 이용함으로써 효율성을 극대화 시킬 수 있으며 많은 클라이언트의 CPU를 사용하므로 대용량 데이터의 고속연산처리가 가능하다. 예를 들어 날씨, 주식, 통계분석과 같이 많은 컴퓨팅 작업을 요하는 작업이 서버에 요청되었을 때 서버는 이 작업을 작은 단위로 분할하여 현재 등록된 클라이언트의 시스템으로 전송하고, 클라이언트 시스템은 전송받은 작업내용을 공유자원을 이용하여 처리한 후 서버에 전송하고 서버는 전송된 결과를 취합하여 작업 요청자에게 전달한다.

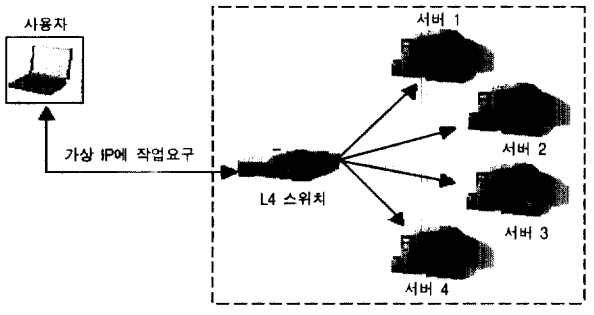


(그림 1) P2P 클러스터링

2.2 H/W 클러스터링

H/W L4 스위치는 클러스터 솔루션(cluster solution)을 구성하기 위한 고기능 스위치로 서버의 부하분배, Cache Redirection, Firewall Load Balancing의 기능을 지니고 있다. L4 스위치의 서버 부하분배는 복수개의 서버를 하나의 가상 IP(virtual IP)로 묶어 사용자에게는 가상 IP만을 알려주고 서버 부하분배기능을 이용하여 네트워크의 트래픽을 가용 서버에 분산하여 전체적인 네트워크 기능을 향상시키는 방식이다. 이러한 부하분배방식은 최대 256개의 가용 서버를 하나의 가상 IP로 묶어 사용자의 서비스 요청이 있을 때 Best Server에 사용자 요청 세션을 열어주고, 실제 서버로 가는 동일한 패킷들은 세션이 끝날 때 까지 실제 서버와 연결을 유지한다. 가상 IP가 아닌 실제 서버의 IP로 서비스가 요청되는 경우 Layer-2/3에서 스위칭이 되며 IP 주소는 바뀌지 않고 MAC주소만 바뀌어 분배된다. 이와 같이 H/W L4 스위치를 이용한 클러스터 방식은 여러 서버들을 한데 묶은 후 이를 가상 IP주소(virtual IP address)로 공유하며, 클라이언트는 이 가상 IP를 이용하여 서비스 신청을

하게 된다. L4 스위치는 전용 하드웨어를 사용함으로 속도와 신뢰성이 좋은 반면 스위치에 문제가 발생할 경우 전체 시스템의 접근이 불가능하고 고가라는 단점을 지니고 있다.



(그림 2) L4 스위치를 이용한 클러스터링

2.3 S/W 클러스터링[7-9]

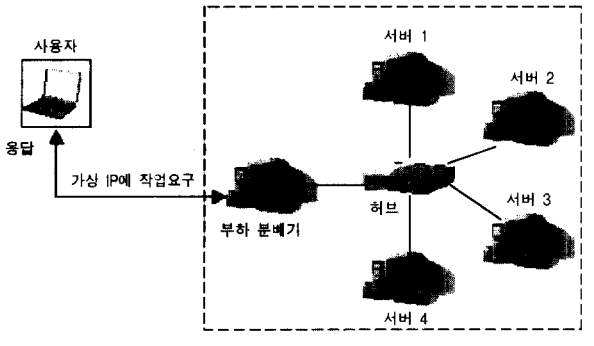
S/W L4 스위치는 소프트웨어적으로 L4 스위치와 동일한 작업을 수행하도록 한 것으로 여러 대의 서버들에 대해 가상의 IP를 할당하고 클라이언트는 이 가상 IP에 작업을 요청한다. 요청은 작업 분배서버에 의해 실제 작업서버로 분배되고 이 서버가 요청에 대한 서비스 작업을 수행한다.

2.3.1 가상 서버의 동작 구조

가상 서버를 구축하기 위하여 클러스터의 실제 서버들은 단일 IP 주소상에 하나의 가상서비스로 보이도록 구성되어야 한다. 이러한 가상 서버 구축방식으로는 네트워크 주소 변환(network address translation)방식과 IP 터널링(IP tunneling)에 의한 구축방식이 있다.

2.3.1.1 네트워크 주소 변환(NAT)

네트워크 주소변환(NAT)방식은 인터넷상의 공인된 IP주소로 들어오는 클라이언트의 프로토콜 헤더를 수정하여 클러스터 내부의 다른 서버들로 연결시키는 방식으로 클라이언트 입장에서는 단일한 서버에 직접 연결된 듯 보이는 방법이다. 이러한 NAT의 특성을 이용하여 서로 다른 IP 주소로 구축된 시스템들을 하나의 IP 주소상의 가상서비스로 구축할 수 있다.



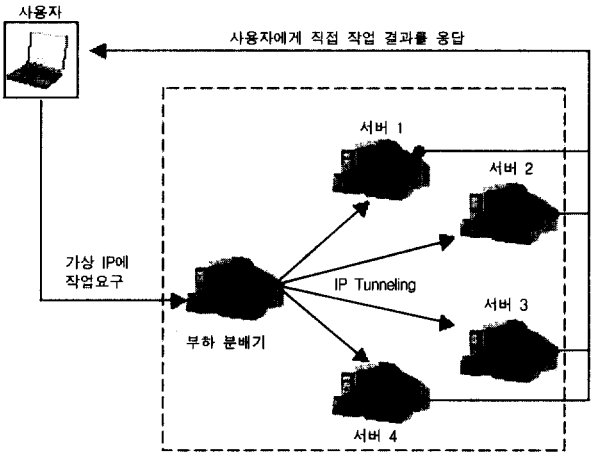
(그림 3) 네크워크 주소변환 (NAT)

(그림 3)에서 보는바와 같이 클라이언트가 클러스터로 구성되어 있는 가상의 서버에 서비스를 요청하면 가상 IP 주소로 향하는 요구패킷은 부하 분산기에 도달하고, 부하 분산기는 패킷의 목적지 주소와 포트번호를 검사한 후 스케줄링 알고리즘에 따라 실제 서버를 선택한다. 부하 분산기는 구성된 접속의 내용을 해쉬 테이블에 저장하고, 실제 서버는 부하분산기로부터 전달된 클라이언트의 요구작업을 실행한 후 결과 패킷을 다시 부하 분산기에 전달한다. 부하 분산기는 해쉬테이블에 저장된 접속내용이 들어온다면 그 패킷의 발생지 주소와 포트를 가상 IP주소의 것으로 변경하여 클라이언트에 전송한다.

이러한 NAT에 의한 가상서버는 TCP/IP를 지원하는 어떠한 운영체제에서도 운용될 수 있고 또한 단지 부하분산기만 실제 IP주소를 필요로 하고 다른 실제 서버들은 자체적인 주소체계를 사용할 수 있다는 장점을 지니고 있다. 반면에 서버노드의 수가 25개 이상 늘어날 경우 실제 서버에 작업분산을 수행하는 부하분산기가 전체 시스템의 병목구간이 될 수 있는 문제점을 지니고 있다.

2.3.1.2 IP 터널링 (IP Tunneling)

IP 터널링은 NAT와 같이 하나의 IP 주소상의 가상 서비스로 구축되어진다. 그러나 NAT와 달리 클라이언트로부터 전달된 요구패킷을 실제 서버의 IP로 감싸서 전달됨으로 실제 서버는 요구된 내용을 처리하고 그 결과를 직접 작업을 요구한 클라이언트에게 돌려줄 수 있다.



(그림 4) IP 터널링

IP 터널링에 의한 서버의 동작은 먼저 클라이언트가 클러스터가 제공하는 서비스에 작업을 요청하면, 요구패킷은 가상 IP주소의 부하분산기에 도달하고, 부하분산기는 패킷의 목적지 주소와 포트를 검사한 후 실제 서버와 접속이 확립되면 해쉬테이블에 접속기록을 추가해 넣는다. 부하 분산기는 요구패킷을 IP 데이터그램에 넣은 후 선택된 서버로 전송한다. 실제 서버는 전달된 패킷을 받으면 그 패킷을

분석하여 요청된 서비스를 처리하고 그 결과를 클라이언트에게 직접 돌려준다.

IP 터널링에 의해 구축되어지는 클러스터의 실제 서버들은 실제의 IP주소를 가질 수 있고 여러 지역에 분산되어 구축되어질 수 있다는 장점을 지니고 있는 반면에 각각의 실제 서버들은 IP encapsulation protocol을 지원해야한다.

2.3.2 가상서버의 작업분배

2.3.2.1 Round-Robin 과 Weighted Round-Robin

Round-Robin 방식은 실제 서버의 접속수나 응답시간을 고려하지 않고 모두 동일하다는 가정하에 순서적으로 작업 서버를 선택하는 방식이다. 따라서 Round-Robin방식은 실제 서버들 사이에 동적인 부하불균형을 초래할 수 있다. 이에 반하여 Weighted Round-Robin 방식은 각 실제 서버에 처리용량에 비례한 가중치를 부여하여 작업을 분배하는 방식이다. 따라서 각 서버들은 가중치에 따라 작업할당 순서가 정해지며 네트워크의 접속은 정해진 순서에 따라 Round-Robin 방식으로 실제 서버들로 보내진다.

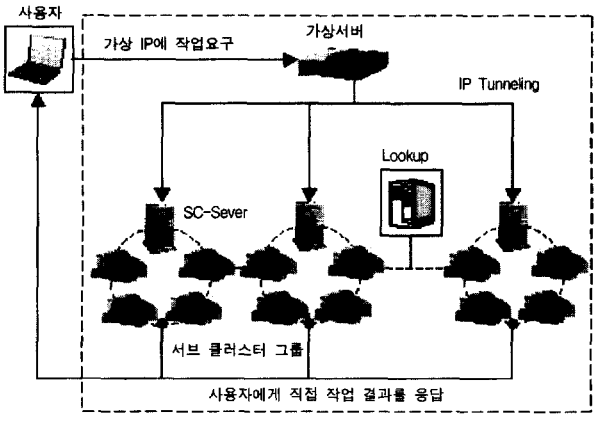
2.3.2.2 Least-Connection과 Weighted Least-Connection

Least-Connection은 동적 스케쥴링 방식의 한가지로 현재의 접속상황을 모니터링하여 접속량이 가장 작은 서버를 선택하여 작업을 할당하는 방식이다. 이와 같은 Least-Connection 방식은 구성된 가상서버가 비슷한 성능의 서버들로 구축된 경우 많은 요구들이 하나의 실제 서버에 집중적으로 보내지는 일이 없기 때문에 부하가 많을 때에도 작업의 분배를 원활하게 진행시킬 수 있다.

Weighted Least-Connection은 각각의 실제 서버에 성능 가중치를 부여하고 동적으로 작업 접속상황을 조사하는 방식으로 서버를 선택한다.

3. 제안한 다중 분산 클러스터 그룹

제안한 다중 분산 클러스터 그룹은 다수의 저가(low price), 저속의 시스템을 대상으로 클러스터 그룹을 생성한다. 클러스터 그룹내의 실제 서버들은 복수개의 시스템들을 단일한 가상 IP에 묶어놓은 서브 클러스터로 구성되며 이중 한 시스템이 서브 클러스터를 대표하는 분산 서버가 된다. 따라서 구성된 복수개의 서브 클러스터를 기반으로 형성된 다중 분산 클러스터 그룹은 복수 사용자로부터 요구되어지는 대규모의 작업에 대하여 부하분배 및 MPP 분산 병렬 컴퓨팅 방식을 이용함으로서 처리효율을 극대화시킬 수 있다. 제안한 다중 분산 클러스터 그룹은 일반 Web Service 작업이외에 Database 관리, 병렬연산을 요구하는 대규모 연산 작업등에서 더욱 효율을 발휘할 수 있다. (그림 5)는 제안한 다중 분산 클러스터의 구성도이다.

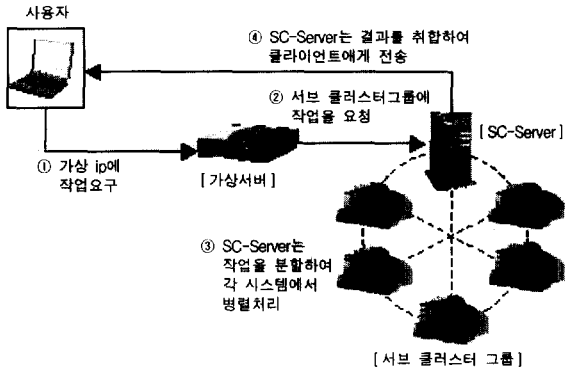


(그림 5) 다중 클러스터 그룹

3.1 다중 분산 클러스터의 동작 방식 및 작업분배

3.1.1 동작방식

다중 분산 클러스터 그룹은 사용자로부터 요청된 서비스가 가상 IP상의 부하 분산기로 전송되어지면 부하 분산기는 패킷의 목적지 주소와 포트번호를 검사하고, 만일 요구된 서비스가 가상 서버 서비스와 맞는다면 작업 할당 방식에 따라 실제 작업을 수행할 서브 클러스터 그룹을 선택한다. 부하 분산기는 서비스 패킷을 IP 데이터그램(IP datagram)안에 넣은 후 선택된 서브 클러스터 그룹에 전송한다. 서브 클러스터 그룹내에 선출된 SC-Server(elected SC-Server)는 전송된 패킷을 풀어 요청된 작업을 분석한다. 분석된 작업의 내용은 병렬처리방식(parallel processing method)에 의해 분리되어지고, 분리된 작업의 내용이 서브 클러스터 그룹을 구성하고 있는 각 시스템을 통해 분산처리 된다. 분산 처리된 결과값은 SC-Server에 의해 취합되며 그 결과값은 서비스를 요구했던 클라이언트에게 직접 전송된다. 이때 가상서버의 부하분산기와 각 서브 클러스터 그룹을 구성하고 있는 시스템들은 실제의 IP를 가질 수 있고 또한 서브 클러스터의 SC-Server는 lookup Server를 통하여 서브 클러스터 그룹의 멤버들에 대한 정보를 지니고 있어야 한다.



(그림 6) 분산 클러스터의 동작방식

3.1.2 작업분배

다중 분산 클러스터그룹의 부하 분산기는 클라이언트로부터 요구되어지는 서비스요구들을 각 서브 클러스터 그룹에 할당하기 위하여 Weighted Round-Robin을 사용한다. 구성되어진 클러스터 그룹은 SC-Agent에 의하여 해당 시스템들의 시스템성능, 자료 전송률, 현재 데이터 처리 여부 등을 분석하고 이 클러스터 특징값(SC_{cn})에 의하여 각 서브클러스터의 가중치값(W_n)이 결정된다.

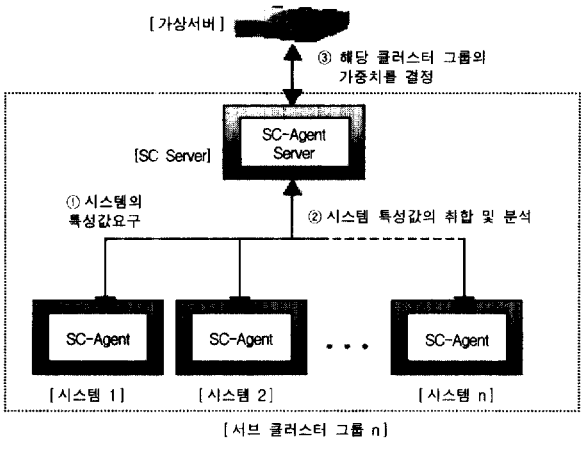
$$SC_{cn} = \sum_{i=1}^{n} (C_i + M_i + N_i)/n \quad (1)$$

식 (1)에서 C_i 는 해당 시스템의 CPU Clock의 값을 M_i 는 가용 Memory의 상태, N_i 는 네트워크 전송량을 나타낸다.

$$W_n = Sort(SC_{cn} \ , \ Job_{vm}) \quad (2)$$

가중치(W_n)는 클러스터 특징값에 현재 수행중인 작업이수{ JOB_{vm})에 의해 결정되어진다.

높은 가중치를 가지는 서브 클러스터 그룹은 더 많은 비율의 서비스 접속요구를 받는다. 따라서 각 서브 클러스터 그룹의 접속수의 비율은 가중치에 비례한다.

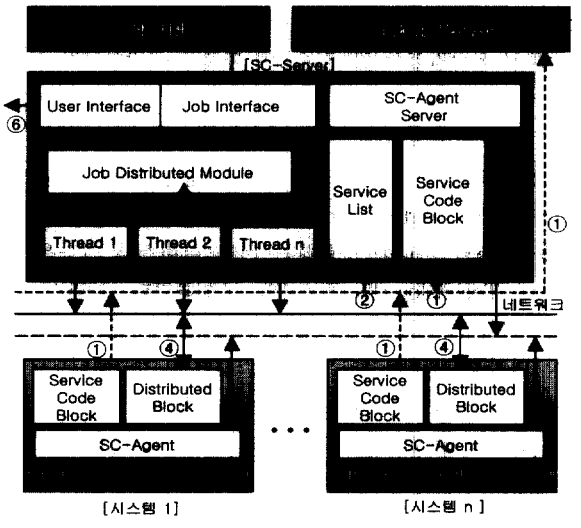


(그림 7) 서브클러스터 그룹의 가중치 결정

3.2 서브 클러스터그룹의 구성 및 분산작업

다중 분산 클러스터그룹을 구성하는 대상은 저가, 저속의 시스템이므로 앞장에서 기술한 클러스터 시스템의 각 노드(node)들처럼 각각의 서버기능을 지니기에는 무리한 조건이 된다. 따라서 일정 수량의 시스템을 동적 서브 클러스터 그룹으로 구성하고 부하 분산기로부터 전송되어온 클라이언트의 서비스 작업을 분산 처리할 수 있는 병렬 컴퓨팅 구조로 구성한다.

(그림 8)에서 모든 서브 클러스터그룹은 전체 서브 클러스터의 서비스 모듈을 통합 관리하는 Lookup Server를 중심으로 구성되어있다. 이 Lookup Server에는 각 서브 클러



(그림 8) 서브 클러스터 그룹에서의 Lookup, Sc-Server 그리고 시스템

스터를 구성하는 시스템들의 서비스 코드 블록(service code block)을 등록 받으며 SC-Server가 요구하는 서비스를 제공하는 역할을 수행한다. 이때 서비스 코드 블록은 해당 시스템에서 처리할 수 있는 서비스의 코드이며, SC-Server는 가상 서버의 부하 분산기로부터 전송되는 작업의 내용을 분할하여 서브 클러스터의 시스템들이 병렬 처리할 수 있도록 작업을 분배하는 역할을 수행한다.

먼저 서브 클러스터 그룹의 각 시스템들은 자신의 서비스 코드 블록을 Lookup Server에 등록한다.(①) 가상 서버부터 해당 클러스터에 작업이 요청되어지면 SC-Server는 Service List 모듈을 통하여 Lookup Server로부터 수행 가능한 서비스의 목록(시스템의 목록)을 전송 받는다. (②), 전송되어진 서비스 목록은 Job Distributed Module로 전달되고, 수행해야할 작업의 내용이 서비스의 목록에 비례하여 분할되어 진다. 분할되어진 각 작업의 내용은 각각 쓰레드를 발생시키고 발생된 쓰레드는 서비스를 수행할 수 있는 시스템과 원격 메소드 호출(remote mothod invocation)을 이용하여 서비스를 제공하는 시스템에서 수행된다.(③) 서브 클러스터 그룹을 구성하는 각 시스템의 Distributed Block은 SC-Server에서 구동된 쓰레드로부터 처리해야할 데이터를 전송 받아 작업을 수행하고(④), 각각의 시스템에서 수행된 작업의 결과값은 SC-Server의 Job Distributed Module로 취합되어진다.(⑤) 취합되어진 내용은 작업을 요청한 사용자에게 직접 전송되어진다.(⑥)

4. 구현 및 고찰

제안한 다중 분산 클러스터 그룹은 웹 환경에서 저성능 시스템을 이용한 고가용성, 고효율 클러스터 시스템이다. 실험의 내용은 서브 클러스터 그룹의 구성과 SC-Server의 분

산작업의 효율에 대하여 실험하였다. 실험을 통하여 제안한 시스템의 확장가능성 및 유용성에 대하여 고찰하고자 한다.

4.1 가상 서버의 구축

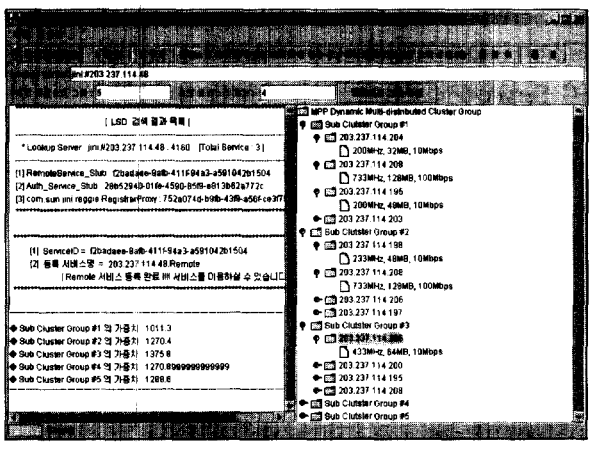
제안한 다중 분산 클러스터 그룹의 상층구조는 리눅스 가상 서버(linux virtual server)를 기반으로 구축되었고 하부의 분산 병렬 처리를 수행하는 서브 클러스터 그룹은 Sun의 Jini Technology를 이용하여 구성하였다. 따라서 서브 클러스터를 구성하는 시스템은 사용 OS에 관계없이 자바 가상 머신을 실행할 수 있는 시스템이면 가능하다. 실험에 참여한 시스템은 총 16대로 Linux와 windows 환경의 시스템이 실험에 참여하였고, 각 시스템은 모두 고유 IP를 가지고 인터넷에 연결되어 있다.

4.2 서브 클러스터의 동적 구성

다중 분산 클러스터 그룹에 참여하는 전체 시스템들은 사용자의 서비스 요구가 요청되었을 때 동적으로 서브 클러스터 그룹을 구성한다. <표 1>은 실험에 참여한 시스템의 목록이다.

〈표 1〉 참여 시스템 목록

No	IP	CPU	RAM	Network	OS
1	203.237.114.48	600	128	10	WinMe
2	203.237.114.194	433	64	10	Linux
3	203.237.114.195	166	32	10	Win98
4	203.237.114.196	200	48	10	Win98
5	203.237.114.197	600	96	10	Linux
6	203.237.114.198	833	128	100	Win2000
7	203.237.114.199	233	48	10	Win98
8	203.237.114.200	433	94	10	win98
9	203.237.114.201	1000	256	100	Win2000
10	203.237.114.202	800	128	100	WinMe
11	203.237.114.203	433	128	10	Linux
12	203.237.114.204	600	128	100	WinMe
13	203.237.114.205	200	62	10	Win98
14	203.237.114.206	600	128	100	Linux
15	203.237.114.207	433	64	10	Linux
16	203.237.114.208	733	128	100	WinMe



(그림 9) 서브 클러스터 그룹의 구성

(그림 9)는 총 16대의 참여 시스템이 5개의 서브 클러스터 그룹을 구성하고 각 구성된 서브 클러스터에 각각 4대씩 배치되는 상황을 보여주고 있다.

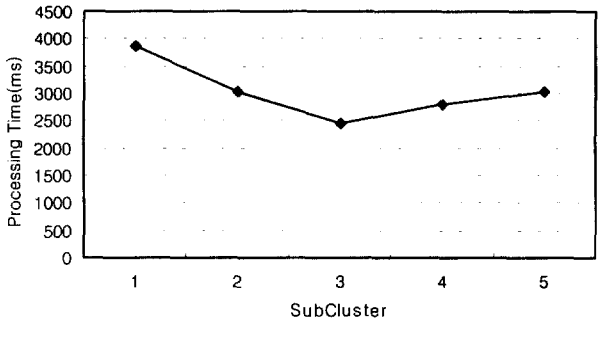
각 서브 클러스터에 배치되는 시스템들은 임의선정을 통하여 무작위 배치된다. 따라서 여러 서브 클러스에 중복 배치되는 시스템이 존재할 가능성이 있다. 구성된 서브 클러스터 그룹은 클러스터 특징값(SC_{cn}), 현재 작업여부에 의해 작업의 우선순위가 결정지어 진다. 구성된 서브 클러스터 그룹은 수행하는 작업이 아직 없으므로 우선순위는 클러스터 특성값에 의해 주어진다. <표 2>은 구성된 서브 클러스터 그룹에 대한 시스템 배치현황과 특성값을 나타낸다.

〈표 2〉 각 서브 클러스터 그룹의 구성시스템 및 특성값 (SC_{cn})과 우선순위

클러스터 그룹	Sub Cluster1	Sub Cluster2	Sub Cluster3	Sub Cluster4	Sub Cluster5
참여 시스템 (203.237.114)	204 208 195 203	198 208 195 203	206 200 195 208	201 195 203 199	198 197 48 196
특성값 (SC_{cn})	1011.3	1270.4	1375.8	1270.89	1288.6
우선순위	⑤	④	①	③	②

4.3 서브 클러스터에서의 작업분배

구성된 다중 분산 클러스터 그룹은 5개의 서브 클러스터 그룹으로 구성되며 각 서브 클러스터 그룹은 부하 분산기로부터 들어오는 클라이언트의 작업요청에 따라 동적으로 최대 4대로 구성한다. 먼저 클라이언트의 작업 요구량에 따른 서브 클러스터의 작업분배조절 능력을 실험하였다. 실험은 800×800 배열연산을 100회 요구하였을 때의 작업분배와 처리완료시간을 측정하였다.



(그림 10) 서브클러스터의 작업분배 및 처리시간

위의 실험에서 3개의 서브클러스터가 연산에 참여하였을 때에는 처리효율이 향상되고 있었지만 4개 이상의 서브클러스터가 작업에 참여하였을 때에는 오히려 작업효율이 감소함을 보여주고 있다. 이는 실제 연산에 소요되는 시간에 비하여 실험데이터를 해당 서브 클러스터로 전송하기 위한 전송시간이 더 많이 소요되어 오히려 처리효율이 저하되어

짐을 확인하였다. 따라서 데이터의 전송방식에 대한 문제를 해결한다면 전체적인 작업의 효율은 더욱 향상될 수 있음을 볼 수 있다.

4.4 부하분배

다음 실험은 클라이언트들의 처리요구가 폭주할 때 해당 다중 분산 클러스터 시스템이 이러한 요구를 서브 클러스터에 적절히 분배하고 전체적인 전송률을 올릴 수 있는지를 실험하였다. 실험의 내용은 1000~4000건의 데이터 처리 요구를 동시에 다중 분산 클러스터에 요청하고 클라이언트의 처리요구를 할당받은 서브 클러스터는 해당 요구를 처리한 후 그 결과를 클라이언트에 직접 전송한다. 실험환경은 위 실험과 동일하다.

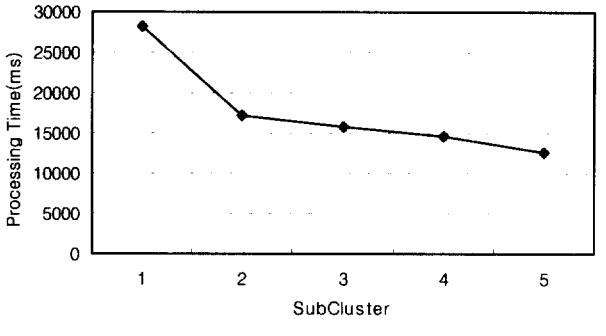
〈표 3〉 서브 클러스터에 의한 처리시간(ms)

Request \ SubCluster	1	2	3	4	5
1000	7466.8	4486.4	4287.4	3979.8	3466.4
2000	4534.8	8603.2	8273.6	7455.2	6727.6
3000	1919.4	12833.4	12552.2	10849.6	9950.0
4000	8188.2	17168.8	15825.4	14500.6	12680.4

<표 3>에서 동시에 처리할 데이터가 1000건인 경우 한 개의 서브클러스터일 때 7466.8ms의 처리시간을 보였고 두 개의 서브 클러스터가 작업에 참여하는 경우에는 4486.4ms의 처리시간을 보였다. 또한 5개의 서브 클러스터가 작업에 참여한 경우에는 3466.4ms의 처리시간을 보임으로써 작업시간이 약 54% 향상됨을 확인할 수 있었다.

위 실험을 통하여 다중 분산 클러스터 그룹 시스템을 사용할 때 전체적인 작업의 효율을 평균 4000건을 기준으로 53%이상의 성능향상을 볼 수 있음을 확인하였다.

(그림 10)은 동시 작업처리량이 4000일 때의 참여 서브 클러스터 수에 대한 처리시간을 나타내고 있다.



(그림 11) 동시처리량이 4000일때의 소요시간

5. 결 론

제안한 다중 분산 클러스터 그룹은 복수개의 시스템이 병렬 컴퓨팅 구조로 구축된 서브클러스터 그룹을 기반으로 구성되어지며 대규모 연산 및 처리요구의 폭주에 대하여 부하의 분산과 병렬 연산을 이용함으로서 처리효율을 향상시킬 수 있었다. 다중 분산 클러스터 그룹을 구축하기 위하여 먼저 상층의 구조는 리눅스 가상서버를 기반으로 구성하였고, 서브 클러스터 그룹은 요구된 작업량에 따라 SC-Server를 중심으로 동적으로 클러스터 그룹을 생성하여 병렬 처리하였다. 실험을 통하여 처리요구가 폭주하였을 때 참여하는 서브 클러스터 그룹에 비례하여 처리의 효율이 더욱 향상됨을 확인할 수 있었다.

향후 연구과제로 대용량의 연산 작업시 데이터의 효율적인 처리방식에 대한 연구와 함께 서브 클러스터 그룹을 구성하고 있는 내부 시스템들의 부하정보를 이용한 작업분배 방식과 서비스 수행중 SC-Server의 고장에 대한 동적 결함복구 방안과 함께 클러스터 그룹이 개방형 네트워크 상에서 구성되어 있으므로 3자 침입에 대한 보안모듈의 개발에 대한 연구가 수행되어야 하리라 사료된다.

참 고 문 헌

[1] R. H. Arpaci-Dusseau et al. "The architectural costs of streaming i/o : A comparison of workstations. clusters, and smps," In Proc. 1998 fourth Int'l sympon High Performance Computer Architecture, pp.99-101, February, 1998.

[2] D. Ridge, "Beowulf : Harnessing the Power of Parallelism in a Pile-of-PCs," In Proc. of IEEE Aerospace conference, pp.79-91, 1997.

[3] H. Wang et al "Computing programs containing band linear recurences on vector supercomputers," IEEE Trans. on Parallel and Distributed Systems, 7(12) : 769-782, August 1996.

[4] M. Nakamura. "New fast algorithms for first-order linear recurrences on vector computers," In Fifth Workshop on compilers for Parallel Computers(Malaga, Spain), pp.167-174, July, 1995.

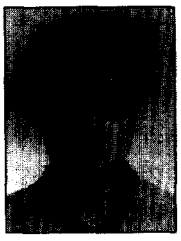
[5] Goddard Space Flight Center. "the HIVE : Highly-parallel Integrated Virtural Environment," http : //newton.gsfc.nasa.gov/thehive/, 1997.

[6] Y. Tanaka, M. Matsuda, "Highly Efficient Implementation of MPI Point-to-Point Communication Using Remote Memory Operations," In Proc. of International Conference on Supercomputing. pp.267-273, 1998.

[7] Ralf S. Engelschall, "Load Balancing Your Web Site : Practical Approaches for Distributing HTTP Traffic," Web Techniques magazine. 3, 5 May, 1998.

[8] Eric Anderson, Dave Patterson "The Magicrouter : an Application of Fast Packet Interposing," http : //www.cs.berkely.edu/ Research/rapid-sweb/SWEB.html, May, 1996.

[9] Om. p. Damani, P. Emerald Chung "ONE-IP : Techniques for Hosting a Service on a Cluster of Machines," http : //cu.utexas.edu/ user/damani. August, 1997.



이 기 준

e-mail : cholee@shinbiro.com

1994년 조선대학교 전산통계학과(이학사)

1997년 조선대학교 일반대학원 전산통계학과 (이학석사)

1998년~현재 조선대학교 일반대학원 전산통계학과 박사과정

관심분야 : 신경망, 패턴인식, 인공지능, 분산 에이젼트 시스템

박 경 우

e-mail : kwpark@www.kjhc.ac.kr

1987년 조선대학교 전산통계학과(이학사)

1994년 조선대학교 일반대학원 전산통계학과(이학석사)

2000년 조선대학교 일반대학원 전산통계학과(이학박사)

1996년~현재 광주보건대학 전산정보처리과 교수

관심분야 : 전문가시스템, 분산에이전트시스템, 지리정보시스템, 지식관리 시스템, 멀티미디어

정 채 영

e-mail : cyjung@mail.chosun.ac.kr

1983년 조선대학교 컴퓨터공학과(이학사)

1986년 조선대학교 일반대학원 전자과 전산전공(공학석사)

1989년 조선대학교 일반대학원 전기과 전산전공(공학박사)

1986년~현재 조선대학교 자연과학대학 수학·전산통계학부 부교수

관심분야 : 영상처리, 신경망, 데이터베이스, 멀티미디어 컨텐츠