한국게임학회 논문지
제4권 제4호 2004년 12월

게임서버의 CPU 사용율 기반 효율적인 부하균등화 기술의 설계 및 구현

*명원식O **한준탁
*동국대학교 컴퓨터공학과, **동해대학교 컴퓨터공학과
*wsmyung@dgu.edu, **okebary513@donghae.ac.kr

Design and Implementation of Game Server using the Efficient Load Balancing Technology based on CPU Utilization

*Myung Won-Shig, **Han Jun-Tak
*Dept. of Computer Engineering, Dongguk University
**Dept. of Computer Engineering, Donghae University

요약

과거의 온라인 게임은 일대일 접속으로 두 사람만이 데이터를 주고받으며 게임 할 수 있었으나 현재의 온라인 게임은 MMORPG라고 해서 수만 명의 사람들이 동시에 접속이 가능하다. 특히 우리나라는 세계 어느 나라에서 찾아 볼 수 없는 네트워크 인프라를 확보하고 있다. 거의 모든 가정에 초고속 인터넷 통신망이 설치되어 있으며 높은 인구 밀도는 이런 인프라의 형성을 가속화하는 것을 가능하게 했다. 하지만 이러한 온라인 게임의 급격한 증가는 제한적인 인터넷의 통신용량에 대하여 트래픽의 증가로 이어지고 온라인 게임이 접속이 불안정해지거나 접속이 다운되는 상태로 이어질 가능성이 높다. 이러한 문제를 해결하기 위해 각 게임 서버를 보다 확충함으로서 해결할 수 있으나 그럴 경우 고비용을 필요로 하게 된다. 본 논문에서는 이러한 문제점을 해결하고자 현재의 온라인 게임에서 사용되고 있는 컨텐츠별로 나누어진 게임 서버들을 지역 클러스터링 형태로 연결하고, 부하 균등화(Load Balancer) 서버로써 특정한 게임 서버의 부하를 감소시키고 게임 서버의 성능 향상과 효율적인 게임 서버 운용을 위하여 부하 균등화 기법을 제안한다. 본 논문에서는 그룹별 각각 다른 서비스를 하고 CPU 사용율의 자원정보를 이용하여 효율적으로 부하를 균등화하는 기법을 제안한다. 각각 서로 다른 게임을 서비스 하는 그룹들은 컨텐츠들에 대한 수정, 삭제, 추가 등 자원 정보 변경으로 인하여 깨어질 수도 있는 자원 정보 일관성을 유지하기 위해 네트워크 파일 시스템에 연결되어 운영된다. 성능 실험을 통해 기존의 RR방식과 LC방식보다 제안한 방식이 각각 12%와 10%의 응답시간 향상을 보여주었다.

Abstract

The on-line games in the past were played by only two persons exchanging data based on one-to-one connections, whereas recent ones (e.g. MMORPG: Massively Multi-player Online Role-playings Game) enable tens of thousands of people to be connected simultaneously. Specifically, Korea has established an excellent network infrastructure that can't be found anywhere in the world. Almost every household has a high-speed Internet access. What made this possible was, in part, high density of population that has accelerated the formation of good Internet infrastructure. However, this rapid increase in the use of on-line games may lead to surging traffics exceeding the limited Internet communication capacity so that the connection to the games is unstable or the server fails. expanding the servers though this measure is very costly could solve this

problem. To deal with this problem, the present study proposes the load distribution technology that connects in the form of local clustering the game servers divided by their contents used in each on-line game reduces the loads of specific servers using the load balancer, and enhances performance of server for their efficient operation. In this paper, a cluster system is proposed where each Game server in the system has different contents service and loads are distributed efficiently using the game server resource information such as CPU utilization. Game servers having different contents are mutually connected and managed with a network file system to maintain information consistency required to support resource information updates, deletions, and additions. Simulation studies show that our method performs better than other traditional methods. In terms of response time, our method shows shorter latency than RR (Round Robin) and LC (Least Connection) by about 12%, 10% respectively.

Key Words: RR(Round Robin), LC(Least Connection), NFS(Network File System)

1. 서 론

현재 우리나라의 인터넷 사용율은 세계 최고 수준을 자랑할 정도로 인터넷 붐이 더욱 확산되고 있다. 이런 분위기를 주도하는 것은 무엇보다 광케이블을 이용한 초고속 정보통신망의 대중적인 보급을 들 수 있으며 기하급수적으로 늘고 있는 PC게임방의 역할도 무시 할 수 없다. 이처럼 전 세계가 인터넷으로 연결되어 국경조차 무의미해지고 있는 지금, 컴퓨터 게임 역시 컴퓨터(오프라인)로만 즐기던 것에서 벗어나 온라인으로 자리를 옮겨가고 있다[15].

하지만 이러한 온라인 게임의 급격한 증가는 제한적인 인터넷의 통신용량에 대하여 트래픽의 증가로 이어지고 온라인 게임이 접속이 불안정해지거나 접속이 다운되는 상태로 이어질 가능성이 높다. 이러한 문제를 해결하기 위해 각 서버를 보다 확충함으로서 해결할 수 있으나 그럴 경우 고비용을 필요로 하게 된다[12].

본 논문에서는 이러한 문제점을 해결하고자 현재의 온라인 게임에서 사용되고 있는 컨텐츠들의 그룹(Group)별 클러스터링 형태를 이용한다. 클라이언트에 대한 트래픽을 좀 더 효율적으로 분산하기 위해 그룹별 내의 게임 서버에 대한 자원 정보를 가지고 부하 균등화를 하고자 한다. 이 기법은 효율적인 게임 서버 운용을 위하여 설계하고 구현한다.

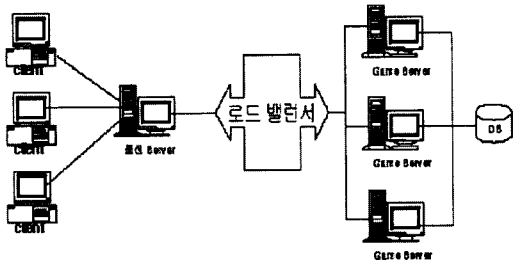
2. 관련 연구

본 절에서는 기존의 온라인 게임에서 게임 서버에 대한 기본 구조와 부하 균등화에 대한 영향을 미치는 성능 요소들에 대해 관찰한다.

2.1 온라인 게임 시스템 구조

온라인 게임에서 게임 서버를 사용하는 가장 큰 이유는 한번에 보다 많은 게이머를 처리할 수 있기 때문이다. 현재의 게임들은 하나의 사회를 이룬다고 할 정도로 대규모로 게이머들의 접속이 가능하게 되었다.

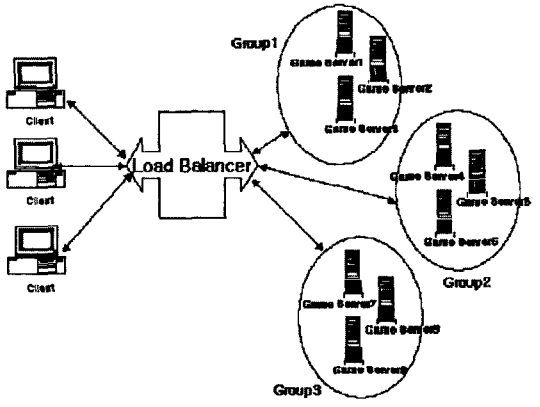
온라인 게임은 클라이언트 부분과 서버 부분으로 구성된다. 즉, 게이머는 컴퓨터상의 클라이언트 프로그램을 구동하여 원격지에 있는 게임 서버에 인터넷을 통하여 접속한 후, 실시간으로 게임을 즐기는 형태이다. 이 구조의 특징은



[그림1] 기존의 온라인 게임 시스템 구조

게임 서버에 실시간으로 접속하는 동시접속자의 수에 따라 네트워크 상에 전송되는 데이터의 양이나 서버에서 처리해야 하는 데이터의 양이 결정된다. 따라서 동시접속자의 수가 점차로 많아지게 되면 네트워크와 서버의 부하량이 지수 함수적으로 증가하여 결국 수용할 수 있는 사용자의 수에 제약이 생긴다[2]. 이와 같은 문제점을 해결하기 위해서는 서버의 확장 등을 통해 게이머들을 분리 운용해야 된다.

[그림1]은 기존의 온라인 게임 시스템 구조를 나타내고 있다. 여기서 통신 서버는 게임 서버 내에 존재하기도 하고 서버 자체를 따로 만들어서 사용하기도 한다. 로드 배런서는 요청되는 클라이언트을 각 게임 서버에 분산하는 역할을 하게 된다. 이 때 분산 스케줄링은 여러 가지 방식을 사용하고 있지만, 대부분 접속자 수에 의해 분산하고 있다. 이 방식은 최소 접속자가 혼자 일 경우 상대방을 기다려야 하는 단점이 발생된다[13].



[그림2] 클러스터링을 이용한 그룹별 시스템 구조

[그림2]는 클러스터링 구조로 동일한 게임들을 그룹별로 복사(Replication)한 것이다. 여기서 각 게임 서버는 분리된 지형적인 배경을 가지며 시간적인 배경은 공유하는 형태를 띠며 게임 캐릭터들의 한 서버로의 집중을 해결하기 위해 다수의 서버에 똑같은 공간 배경을 구현한 후, 이를 클러스터링 기법으로 연결하여 그룹화한 것이다. 로드 밸런서에 의해 특정한 게임 서버의 집중을 분산시키는 방법을 채택하고 있다. 이 방식은 특정한 그룹에 대한 새로운 데이터베이스가 갱신되었을 때마다 각 게임 서버들의 내용도 알맞게 갱신하는 동기화 과정이 필요하다. 이것은 동적인 웹 서비스가 증가하는 경우 상당한 오버헤드로 작용될 수 있다[14].

2.2 부하 균등화에 영향을 갖는 성능 요소

기존의 온라인 게임에 있어 부하 균등화 방식은 초기에 접속자가 적을 때는 한 대만 놓고 운용하다가 접속자가 증가함에 따라 같은 서버를 내부 네트워크로 묶어 동일한 역할을 하게 한다. 이것은 서버의 용량이 증가되므로 편리하다. 이런 경우 특정 서버에 부하가 집중되지 않도록 각 서버의 접속량 등을 파악해 각 서버에 적당히 부하를 분산시키는 기술을 부하 균등화(Load Distribution)이라 한다[2].

부하 균등화 방식은 특정한 서버에게만 게이머들이 집중하는 현상을 막기 위해서 게이머들이 접속을 하면 게이머의 아이디와 패스워드를 확인해서 로그인을 한 후 각기 서버에 균일하게 게이머을 분배하게 된다.

로드 밸런서에서 게임 서버에 대한 자원 정보를 모니터링 하는 도구는 여러 가지 성능 요소들을 관찰하여야 한다. 성능을 반영하는 많은 요소들 중에서 부하 균등화에 영향을 주는 성능 요소는 현재 사용 가능한 물리적 메모리(Physical Memory)와 CPU 사용율, 그리고 네트워크 사용율이다. 여기서 네트워크 사용율은 접속자 수이다. 이 정보들을 이용하여 개별적인 게임 서버의 상태를 모니터링 할 수 있다. 본 논문에서는 네트워크 사용율과 메모리 사용율을 배제하고 CPU 사용율을 가지고 부하 균등화하는 기법을 제안하고 구현한다.

2.3 부하 분배 스케줄링 알고리즘

부하 분배 스케줄링에 이용되는 방법 중 라운드 로빈(Round Robin) 방식은 클러스터로 구성된 모든 웹 서버들이 동등하게 한번씩 돌아가며 요구에 대하여 서비스하는 방식이다. 이는 각 웹 서버의 서비스 처리 용량과 관계없이 서비스되므로 부하 불균형이 발생할 수 있는 단점이 있다. 이러한 문제점을 고려하여 라운드 로빈 스케줄링과 유사하나 클러스터에 속한 웹 서버들의 처리 용량에 따라 가중치를 둔 뒤 연결하는 가중기반 라운드 로빈 스케줄링 방식이 있다. 또한 연결된 사용자 요구 수가 제일 적은 곳을 우선적으로 클라이언트 요구를 전달하는 최소 연결(Least Connection) 스케줄링 방식이 있고, 각각의 실행 웹 서버에 성능 가중치를 부여하고 동적으로 작업 접속 상황을 조사하는 방식으로 실제 웹 서버를 선택하는 가중치 기반 최소

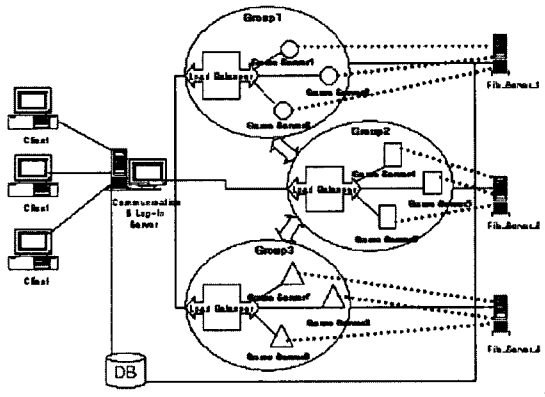
연결(Weighted Least Connection) 스케줄링 방식이 있다.

이러한 부하 분산 기법들을 이용할 경우 하나의 IP 주소로 서비스가 가능하며 게임 서버의 결합 발생 시 나머지 게임 서버들에 부하를 분산시킴으로써 결함을 허용할 수 있는 장점이 있다. 부하 분배기에서 일어날 수 있는 병목 현상은 IP 터널링이나 직접 라우팅을 통해 서비스를 하는 게임 서버와 클라이언트가 직접 통신함으로써 해결될 수 있다 [11].

3. 시스템 구조와 제안된 부하 균등화 기법

본 논문에서 제안하는 효율적인 부하 분산 기법은 기존의 그룹별 게임 서버 시스템 구조([그림3] 참조)에 대한 자원 정보를 가지고 클라이언트 요청을 분산하는 기법을 제안한다.

[그림3]은 클라이언트 수가 폭발적으로 증가함으로써 이에 대처하기 위해 게임 서버를 클러스터링 구조로 접속 수에 따라 부하 균등화하는 방식을 채택하고 있다. 따라서 현재의 온라인 게임은 컨텐츠별(각 게임별)의 게임 서버들을 하나의 클러스터링 기법으로 하고, 여기에 로드 밸런서를 같이 둠으로써 기존의 방식보다는 로드 밸런서의 역할이 감소되고 균등하게 각 서버에게 게이머들을 분배할 수 있다.



[그림3] 제안된 온라인 게임 시스템 구조

[그림 3]은 통신 서버와 로그인 서버를 통합하여 하나의 서버로 최초 접속 게이머에 대한 로그인과 기존 게임에 대한 등급 및 점수와 기타 서비스를 관리한다. 또한, 부하 분산기가 관리하는 게임 서버들로 구성되어 있고, 게임 서버들은 컨텐츠(게임 별)의 종류에 따라 다시 그룹화 되었다. 한 그룹의 게임 서버들은 하나의 네트워크 파일 시스템에 링크되어 있고, 동일한 컨텐츠를 서비스한다. 부하 분산기가 관리하는 게임 서버와 그룹별 수는 각기 다를 수 있다. 특정한 그룹에 대한 새로운 데이터베이스가 갱신되었을 때마다 각 게임 서버들의 내용도 알맞게 갱신하는 동기화 과정이 필요하기 때문에 이를 효율적으로 관리하기 위해 네트워크 파일 시스템(NFS)으로 구성한다. 온라인 게임을 위한 일반적으로 다음과 같이 7가지 기능을 지원할 수 있어야 한다.

① 게이머와의 접속과 사용자 인증을 위한 로그인 기능
② 게이머의 캐릭터 특성과 게이머의 개인정보를 관리하는 데이터베이스 관리 기능
③ 게이머와의 패킷 송수신을 위한 통신 기능
④ 클라이언트 프로그램이나 자료 변경 시 자동 전송을 실시하는 다운로드 기능
⑤ 맵의 확장 시 이를 지원하고 캐릭터의 맵간 이동을 허용하는 확장 기능
⑥ 게이머의 명령을 게임 룰에 따라 해독하여 클라이언트에 전달하는 게임처리 기능
⑦ 특정한 게임 서버에 게이머가 집중되지 않도록 적당하게 분배하는 부하 분배 기능

본 논문에서 제안한 부하 균등화 기법은 위 7가지 기능을 기본적으로 네 개의 독립적인 서버가 분담하여 맡도록 하였다. 즉, 통합적인 데이터베이스의 관리를 담당하는 데이터베이스 서버 및 게임 처리만을 담당하는 게임 서버, 클라이언트의 접속과 인증 및 다운로드를 담당하고 클라이언트를 원하는 게임 서버로 접속시켜주거나 패킷 송수신을 담당하는 통신 및 로그인 서버를 하나의 서버로 구성하였다. 또한 게이머가 원하는 게임 서버에 접속하고자 할 경우 클러스터링으로 구성된 게임 서버들에게 균등하게 게이머을 분배 할 수 있도록 로드 밸런스 서버를 구성하였다.

제안한 부하 균등화 시스템의 기본 동작은 다음과 같다.

① 게이머는 게임을 하기 위해 자신의 컴퓨터상에서 클라이언트 프로그램을 구동시켜 로그인/통신 서버와 접속 한다
② 로그인/통신 서버는 접속된 게이머의 인증과 자동 다

운로드를 위해 데이터베이스 서버에 접속한다.

③ ②를 마치면 게이머에 데이터베이스 서버로부터 얻은 사용자 정보를 전달한다.

④ ③후 로그인/통신 서버는 게이머가 원하는 게임 그룹의 로드 밸런스 서버에 연결된다.

⑤ 로드 밸런스 서버는 자기가 관리하고 있는 게임 서버들 중 가장 적은 CPU 사용율을 가지고 있는 게임 서버에 게 연결한다.

⑥ 게임을 종료할 경우, 게임 서버는 게이머가 정한 캐릭터의 모든 정보를 데이터베이스 서버에 저장한다.

⑦ 게임 배경을 이동할 경우, 로그인/통신 서버는 게임 서버에서 제공하는 캐릭터의 데이터베이스 정보를 전달받아 게이머가 원하는 게임 그룹에 연결한 후 ⑤의 행위를 지속한다.

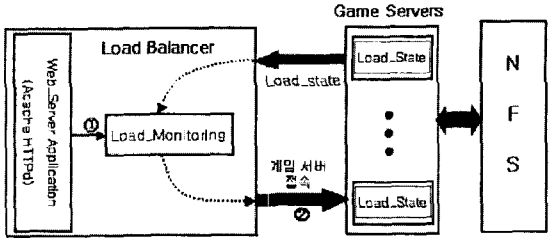
기존 온라인 게임(그림1, 그림2)의 그룹별 클러스터링의 부하 분산의 기본적인 스케줄링 방식은 접속 수에 따라 가장 적은 접속 수에 게이머의 요청을 처리한다. 이는 접속 수가 가장 적은 경우, 한 명도 접속이 되어 있지 않으면 최초의 접속자로서 상대방이 접속하기를 기다려야 하는 단점이 발생한다. 이는 사용자 입장에서 대기 시간(wait time)이 필요함을 알 수 있다.

본 논문에서는 이러한 문제점을 좀 더 효율적으로 해결하기 위해 각 그룹 내의 게임 서버에 대한 자원 정보를 가지고 평균값을 가진 게임 서버에게 클라이언트 요청을 처리하고자 한다. 본 논문에서 자원 정보란 각 게임 서버에 대한 CPU 사용율을 말한다.

3.1 제안한 시스템 구조의 프레임워크

[그림4]는 로드 밸런서와 게임 서버들 간의 모듈 관계를 보여 주고 있다. 이는 특정한 한 그룹 내의 프레임 워크이다. 다른 그룹도 동일하다. Web_Server Application 모듈은 Apache로 구동되며 게임 사이트의 HTTP_데몬이다. 게이머는 HTTP_데몬을 통해서 접속하며, 게임 진행은 해당 게임을 가지고 있는 게임 서버로부터 응답을 받는다.

①과 ②는 게이머가 게임 사이트 접속 후 로그인이 된 다음 해당 게임 서버를 접속 하는 경로를 보여 주고 있다. Load_Monitoring 모듈은 게임 서버들의 자원 정보인 CPU 사용율을 가지고 있다. 이 때 각 그룹 내 게임 서버들은 주기적으로 Load_Monitoring 모듈에게 Load_State(CPU 사용율) 보고해야 한다. 주기적이란 실험 값 75[㎲]이다. Load_Monitoring 모듈은 각 게임 서버들의 CPU 사용율이 가장 작은 게임 서버를 선택 한 후 게이머를 선택된 게임 서버에게 할당해주는 역할을 하게 된다. 아래의 식은 한 그룹 내의 게임 서버에 대한 자원 정보를 가지고 평균값을 구하는 식이다.



[그림4] 로드 밸런서와 게임 서버들간의 프레임워크

$$AVG = \frac{\left(\sum_{i=1}^{n} ResourceInfo\right)}{\#of\ GameServer} \quad (1)$$

$$Selected\_GameServer = Min(|ResourceInfo_i - AVG|) \quad (2)$$

ResourceInfo : 각 게임 서버에 대한 CPU 사용율 ($0 \le ResourceInfo \le 1$), Load_State와 동일

AVG : 그룹 내의 CPU 사용율에 대한 평균값

i : 그룹 내의 게임 서버 고유 번호

# of GameServer : 그룹 내의 게임 서버의 수

Selected_GameServer : 요청을 처리할 선택된 게임 서버

식 (1)과 (2)는 한 그룹 내에서 게이머의 요청을 분산하기 위한 적절한 게임 서버를 선택하는 공식이다. 먼저 그룹 내의 각 게임 서버에 대한 자원 정보(CPU 사용율)를 합한 다음 그룹 내 게임 서버 수로 나눈 값이 해당 그룹 내의 평균값이다. 평균값을 가지고 실제 게임 서버를 선택하기 위해서는 각 게임 서버의 자원 정보와 평균값과의 차이점이 가장 적은 게임 서버를 찾아낸다. 선택된 게임 서버가 게이머 요청을 처리하는 게임 서버가 된다. 예를 들어 〈표1〉과 같이 한 그룹 내의 게임 서버의 자원 정보가 있다고 가정하자.

	CPU 사용율
게임 서버 1	35
게임 서버 2	50
게임 서버 3	60

〈표1〉 게임 서버의 자원 정보(CPU 사용율)

위의 (1)식에 의해 평균값(AVG)는 48이다. 식 (2)에 의해 48을 가지고 각 게임 서버와의 차이를 구하면, 게임 서버 1은 13이고 게임 서버 2는 2이다. 게임 서버 3은 12의 값을 가진다. 따라서 그룹 내의 자원 정보를 모니터링 하는 Load_Monitoring 모듈은 게이머의 요청을 게임 서버 2에게 전달하게 된다. 만일 Selected_GameServer의 값이 동일한 여러 게임 서버가 발생하면 자원 정보 값이 가장 작은 게임 서버에게 전달한다. 또한, 자원 정보 값이 동일한 게임 서버가 여러 개 발생되면 전통적인 부하 분산 스케줄링 방식인 라운드 로빈(Round Robin) 방식으로 게임 서버를 선택하게 된다. CPU 사용율이 80% 넘으면 해당 게임 서버는 부하 분산 대상에서 제외된다.

4. 실험 및 성능 평가

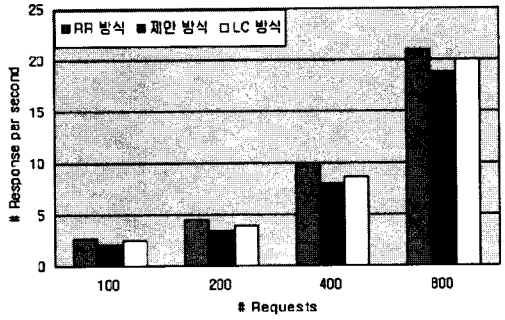
본 장에서는 본 논문에서 제안된 모델의 성능 실험을 측정하기 위해 WOW Linux 7.0 운영체제에서 네트워크 시뮬레이터인 NS2.1b8a를 사용하였다. 시스템 사양은 Pentium III 1GHZ, RAM 256 MB, HDD 30GB로 구성하였다. 실험에 사용된 장비는 〈표 2〉과 같다. 또한, 제안된 시스템 구조에서의 모든 게임 서버 및 로드 밸런서는 동일한 사양을 사용하였다. 성능 평가를 위해 사용한 임의의 로그 파일을 표본 추출하여 Data Set으로 이용하였고, 실험 결과 비교는 앞에서 언급한 RR 스케줄링 방식과 LC 스케줄링 방식, 그리고 제안한 방식에 대해 실험하였다.

장비 이름	CPU	RAM	HDD
Load Balancer	PentiumIII 1GHz	256MB	30GB
Game Server1	PentiumIII 1GHz	256MB	30GB
Game Server2	PentiumIII 1GHz	256MB	30GB
Game Server3	PentiumIII 1GHz	256MB	30GB

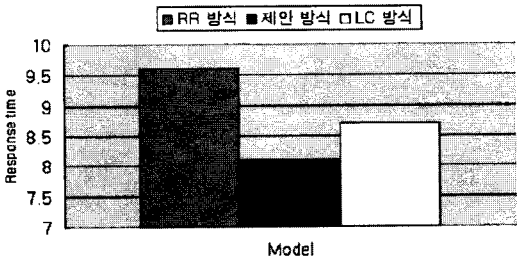
〈표 2〉 실험에 사용된 장비(그룹별 동일)

4.1 실험 결과

기존의 RR 스케줄링 방식과 LC 스케줄링 방식, 그리고 제안한 시스템 구조에 대해 요청 수에 대한 응답시간을 비교 실험하였다. 실험에 사용된 요청 수는 100개, 200개, 400개, 800개의 입력 데이터로 구성하여 실험을 하였다.



[그림5] 요청 수에 따른 응답시간 실험 결과



[그림6] 지수 분포 모델

[그림5]은 요청 수 100개, 200개, 400개, 800개에 따른 응답시간(Response time) 실험 결과이다. 처음의 요청 수가 100개 일 때, RR 방식은 2.7초이고, LC 방식은 2.4초, 제안한 방식은 2.1초이다. 그리고 요청 수가 800개 일 때, RR 방식은 21초, LC 방식은 20초, 제안한 방식은 18.8초이다. 전체적으로 제안한 방식이 RR 스케줄링 방식과 LC 스케줄링 방식 보다 응답시간 향상 되는 것을 알 수 있었다.

[그림6]과 같이 응답시간에서도 지수 분포 모델로 실험 했을 때 RR 방식은 평균 9.6초, LC 방식은 8.7초에 반해 제안한 방식은 8.1초로 각각 12%와 10% 정도로 더 짧다는 것을 알 수 있었다.

5. 결론 및 향후 과제

본 논문에서는 기존의 시스템 구조에서 사용하던 접속자 수로 부하 분산하는 방식을 자원 정보로 부하 균등화하는 방식을 제안하였다. 이는 게임별로 그룹화 된 시스템 구조에서 최초 접속자로 접속되는 것을 최소화하고, 사용자의 대기 시간을 단축시킬 수 있다. 실험에 있어 시뮬레이션을 이용하였기 때문에 실제 물리적인 온라인상에서는 제안한 기법은 온라인에 적용 시 요청에 따른 사용자의 대기 시간(wait time)이 효과적으로 단축되어 보다 빠른 서비스가 가능할 것으로 기대된다. 전체적으로 제안한 시스템의 장점으로는 로드 분산을 효율적으로 할 수 있으며, 가격 대 성능 향상을 볼 수 있다. 반면에 단점으로는 로드 밸런스 서버가 다운되었을 때 해당 그룹의 게임 서비스가 마비될 수 있는 것이다. 따라서 앞으로의 연구 과제로 로드 밸런스 서버가 다운되었을 때 전환 가능한 보조적인 역할을 구현하는 방법을 고려 중이다.

6. 참고 문헌

[1] Dongeun Kim and Cheol Ho Park and Deayeon Park, "Request rate adaptive dispatching architecture for scalable Internet server," Cluster Computing, 2000. Proceedings. IE EE International Conference, pp.289-296, 2000.

[2] Canal, R. and Parcerisa, J. M. Conzalez and A., "Dynamic cluster assignment mechanisms," High-Performance Computer Architecture, 2000. HPCA-6. Proceedings. Sixth International Symposium, pp.133-142, 1999.

[3] Beowulf Project, http://www.beowulf.org.

[4] A. Wong and T. Dillon, "Load balancing to Improve Dependability and Performance for Program Objects in Distributed Real-time Cooperation over the Internet," The 3rd IEEE International Symposium on Object-Oriented Real-time Distributed Computing, Mar., 2000.

[5] V. Carellini and M. Cloajanni and P. Yu, "Redirection Algorithms for Load Sharing in Distributed Web-server Systems," Proceedings of the 19th IEEE International Conference on Distributed Computing Systems, pp. 528-535, May, 1999.

[6] M. Garland, S. Grassia, R. Monroe and S. Puri, "Implementing Distributed Server Groups for the World Wide Web," Tech. Rep. CMU-CS-95-114, Carnegie Mellon University, School of Computer Science, Jan. 1998.

[7] C. S. Yang and M. Y. Luo, "Efficient Support for Content_based Routing in Web Server Clusters," In Proc. of 2th Symposium on Internet Techmologies&Systems, Colorado, USA, October 11-14, 1999.

[8] V. Cardellini, E. Casalicchio, M. Colajanni, M. Mambelli, "Web Switch Support for Differentiated Services," ACM Performance Evaluation Review. Selected from Performance and Achitecture of Web servers Workshop. 2001.

[9] Vivek S. Pai, M. Aron, G. Banga, M. Svendsen, P. Druschel, W. Zwaenepoel and E. Nahum, "Locality-Aware Request Distribution in Cluster-based Network Services," ACM 8th, ASPLOS Oct. 1998.

[10] M. Colajanni, P.S Yu, V. Cardellini, "Dynamic bad balancing in geographically distributed heterogeneous Web-servers," Proc. of 18th IEEE Int'l,. Conf. on Distributed Computing Systems(ICDCS'98), pp.295-302, Amsterdam, The Netherlands, May 1998.

[11] 명원식, 장태무, " 웹 서버 클러스터에서 내용 기반 으로한 부하 분산 기법," 정보처리학회논문 지 A, 제10-A권 6호, Vol. 10-A, No 6, pp.729-736, Dec. 2003.

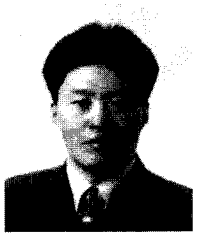
[12] "기술 수준을 중심으로 한 게임 산업의 업계 실태 조사 연구," 동국대학교 산학기술협력센터,2002.2

[13] 명원식,유행석,홍동철,한준탁,장태무,"웹 서버 클러스 터를 이용한 온라인 게임에서의 효율적인 부하 분산 에관한연구," 한국게임학회,2003동계학술대

회,pp.279-282
[14] 전성기,한준탁,명원식,유행석,김수성,장태무,"온라인 게 임에서 동적 부하 분산 시스템," 한국게임학회,2003하 계학술대회,pp.281-285
[15] 유행석,명원식,홍동철,한준탁,장태무,"온라인 게임에서 바로가기 아이콘에 관한 연구," 한국게임학회,2003동 계학술대회,pp.221-224

명 원 식

1996년 안양대학교 컴퓨터공학과 졸업(학사)
1998년 안양대학교 전산정보학과 졸업(공학석사)
2004년 동국대학교 컴퓨터공학과 졸업(공학박사)
관심분야 : 분산 및 병렬처리, 클러스터 웹 서버, 컴퓨터 구조, 온라인 게임 등



한 준 탁

1989년 순천향대학교 전자계산학과 (공학사)
1994년 한양대학교 전자계산학과(공학석사)
2003년 동국대학교 컴퓨터공학과(공학박사)
2004년 2월 동해대학교 정보전산소 소장
1995년 3월~ 현 동해대학교 컴퓨터공학과 조교수 재직 중
관심분야 : 분산 및 병렬 컴퓨팅, 입출력 시스템, 온라인 게임서버 등