2003년 대한전자공학회 하계종합학술대회 제26권 제1호

대용량 온라인 게임 서버를 위한 부하 평형 기법

손민규, 성영락, 오하령, 안현식, 김정윤*
국민대학교 전자정보통신공학부
*재미창조

A Load Balancing Technique for a Massively Multiplayer Online Game Server

Min-gyu Son, Yeong-rak Seong, Ha-ryoung Oh, Hyun-sik Ahn, Jeoung-youn Kim*
School of Electrical Engineering, Kookmin University
Interesting & Creative*
E-mail : sonmg00@empal.com

Abstract

클라이언트-서버 구조의 온라인 게임에서 서버의 역할은 매우 중요하다. 그러나 서버에 너무 많은 사용자가 동시에 접속할 경우에는 과도한 부하로 인하여 안정적인 서비스를 제공하지 못하는 경우가 발생할 수 있다. 이러한 문제를 해결하기 위하여 여러 대의 분산 서버를 활용한 방법이 사용되고 있다. 분산 서버 구조에서는 부하가 특정 서버로 집중되는 것을 막는 것이 전체 시스템의 성능 향상을 위하여 필요하다. 이를 위하여 부하 평형 기법을 사용하고 있으며 관련 연구도 활발히 진행 중이다.

본 논문에서는 분산 서버를 이용한 온라인 게임 서버의 구조를 설계하고, 대량 사용자의 제약을 극복하기 위한 하나의 방안으로 다중 서버 시스템에서 실행되는 게임 서버 프로세스들에 대한 부하 평형 기법을 제시한다. 제안된 기법에서는 전체 게임을 소규모의 여러 게임 세상(Game World)으로 나누고 서비스가 진행되는 상황에서 게임 세상을 게임 서버들 사이에서 이동시킨다.

I. 서 론

최근 네트워크 기술의 발달로 초고속 인터넷이 확산되면서 동시 접속자 수가 수만 명에 달하는 온라인 게임 서비스가 증가하고 있는 추세에 있다. 대부분의 온라인 게임 서버는 클라이언트-서버 구조를 사용하고 있다. 그러나 게임 서버에서는 접속된 모든 클라이언트들의 정보를 받아 처리하므로 너무 많은 클라이언트가 동시에 접속하여 서비스를 받을 경우에 성능이 떨어지는 단점이 있다.

본 논문에서는 대량의 사용자가 게임 서버에 동시에 접속되었을 때의 제약을 극복하기 위한 하나의 방안으로 다중 서버 시스템에서 실행되는 게임 서버 프로세스들에 대한 부하 평형 기법을 제시한다.

II. 관련 연구

2.1 온라인 게임

최근 서비스 중인 대부분의 온라인 게임은 별도로 제공되는 클라이언트 프로그램을 가지고, 초고속 인터넷 서비스 등의 유 무선 통신망을 이용해 게임 서버에 접속하여, 가상의 공간에서 다수의 다른 사용자들과 커뮤니티를 형성하며 정해진 게임 룰에 따라 문제를 해결해 나가는 방식으로 구성되어 있다. 이러한 종류의 온라인 게임을 MMOG (Massively Multiplayer Online Game) 또는 MMORPG (Massively Multiplayer Online Role-Playing Game) 라고 한다[1].

대용량 온라인 게임은 서버에 수천의 사용자 클라

이언트들이 접속한 상태에서 정해진 데이터를 주고받으면서 게임을 진행하므로 서버 측에 부하가 집중된다. 이러한 이유로 서비스의 품질은 서버 시스템의 네트워크 대역폭과 CPU 의 연산 처리 능력에 의해 좌우되는 특성이 있다.

오늘날의 온라인 게임들은 서버의 구성 형태에 따라 크게 셋으로 나눌 수 있다. 첫째는 단일 서버 구조이다. 단일 서버 구조에서는 가상공간 전체를 하나의 서버가 관리한다. 이 구조는 설계와 개발에는 용이하나, 하나의 서버가 감당할 수 있는 클라이언트의 수에 한계가 있다. 이를 극복하기 위해서는 고성능의 서버를 사용해야 하기 때문에, 비용이 많이 들고 서버의 확장이 어렵다는 단점이 있다.

둘째는 중복 서버 구조이다. 이 구조에서는 독립적으로 운영되는 여러 개의 단일 서버를 운영한다. 즉, 동일한 가상공간을 각각의 서버에 복제하여 독립적으로 관리하는 구조이다. 따라서 서버들 간에 사용자들의 게임 정보가 호환되거나 이동하는 것이 불가능한 구조이다.

셋째, 지역 서버 구조는 전체의 가상 세계를 여러 개의 지역으로 분할하여 지역 서버가 하나의 분할된 지역을 관리하도록 한 구조이다. 게임의 기획 시에 가상공간을 분할하는 경우이며, 따라서 지역 서버 간에 동기화를 해야 할 필요가 없어 개발이나 운영에 장점이 있다. 하지만 확장 능력에 한계가 있으며, 사용자는 인접한 지역 서버 사이를 자유롭게 넘나들 수 있으나, 지역 내의 물체는 이동하기 어렵다는 단점을 가진다[2].

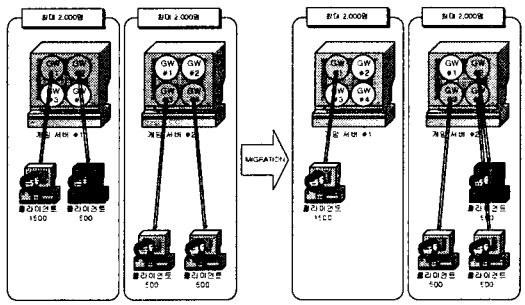
2.2 부하 평형

부하 평형은 다중 프로세서 환경에서 프로세서의 부하가 균등하게 배분되도록 각 프로세서에 작업들을 할당하는 것을 말한다. 부하 평형은 병렬 처리 시스템이나 네트워크로 상호 연결된 시스템 구조 내의 서로 다른 프로세서 사이에서 이루어질 수 있다. 또한 분산 서버 구조 내에서 부하가 많은 서버의 작업을 부하가 적은 서버로 이동시켜 서버간 부하의 불균형을 감소시키고 부하를 고르게 분배하는 것을 말하기도 한다[3].

본 논문에서는 여러 서버에서 다수의 게임 세상을 관리하는 지역 서버 구조에서의 부하 평형 기법에 대해 다룬다. 일반적으로 이런 구조에서는 하나의 게임 세상이 하나의 프로세스로 관리된다. 또한 특정 서버가 사용자 용량의 한계에 도달하면 그 서버에서 배분된 게임 세상 프로세스들 중 일부를 부하가 적은 서버로 이동시켜 부하 평형을 시도한다. 이때 부하를 이동시키는 시점에 따라 오프라인 방법과 온라인 방법으로 구분할 수 있다. 오프라인 방법이란 일정 시간 동안 게임 서비스를 중지한 상태에서 수동으로 게임 세상을 이동시키는 방법이다. 현재 서비스되고 있는 대부분의 온라인 게임들이 이러한 방법을 사용하고 있다. 반면 온라인 방법에서는 게임 서비스가 유지되고 있는 상태에서 게임 세상이 이동된다. 그러므로 시스템에 재해나 오류가 없으면 언제나 게임 서비스가 유지 될 수 있다는 장점이 있다. 또한 서버의 증설 시에도 서비스의 유지가 가능하다. 본 논문에서는 온라인 부하 이동에 기반을 둔 부하 평형 기법을 제안한다.

III. 부하 평형 기법

본 논문에서 제안한 부하 평형 기법에서 각 게임 서버는 모든 게임 세상에 일대일 대응하는 게임 프로세스들을 가진다. 그러므로 게임 서버의 개수가 n, 게임 세상의 개수가 m 이면, 전체 시스템에는 m*n 개의 게임 프로세스가 존재한다. 그러나 이중에서 동작 상태의 프로세스의 개수는 항상 m 개로 유지된다. 즉, 특정 게임 세상에 관련된 n 개의 프로세스 중에서 오직 하나만이 동작 상태이다. 예를 들어 그림 1(a)의 경우, 서버의 개수는 2, 게임 세상의 개수는 4 이므로, 전체 8 개의 게임 프로세스가 존재하지만, 게임 세상 1, 2 는 서버 1 에서, 게임 세상 3, 4 는 서버 2 에서만 동작 상태로 운영된다.



(a) (b)

그림 1. 제안된 부하 평형 기법

본 논문에서는 게임 서버의 부하를 접속된 클라이언트 수에 비례하는 것으로 정하였다. 이는 게임 진행에 필요한 메시지들이 서버에서 클라이언트로 전달될

때 방송(broadcast)의 형태로 전달되기 때문이다. 또한 최대 부하는 동시 접속 클라이언트 수를 2000 명으로 가정하였다. 즉, 어떤 서버에서 동작 상태인 모든 게임 세상 프로세스에 접속된 클라이언트 수의 합이 2000 이 되면 부하가 높은 것으로 판단하여 부하 평형을 시도한다.

부하 평형은 게임 세상을 다른 서버로 이주시키는 방법으로 이루어진다. 즉, 부하가 높은 서버 내의 몇몇 프로세스들을 다른 서버로 이주시키는 것이다. 예를 들어, 그림 1(a) 와 같이 프로세스들이 동작 중인 상태에서, 게임 세상 1 에 1500, 게임 세상 2, 3, 4 에 각각 500 개씩의 클라이언트가 접속할 경우, 서버 1 이 최대 부하에 도달하여 부하 평형이 시도된다. 그 결과 그림 1(b) 에서와 같이, 서버 1 에는 게임 세상 1 만 운영되어 총 1500 개의 클라이언트, 서버 2 에는 나머지 게임 세상들이 운영되어 총 1500 개의 클라이언트가 접속하게 된다.

이 과정에서 중요한 것은 서비스의 중단을 최소화하는 것이다. 즉, 게임 세상이 이주하는 과정에서 필연적으로 서비스의 중단이 생기는데 이를 최대한 줄여야 사용자의 불편을 최소화할 수 있다. 이러한 목적 하에 본 논문에서는 다음과 같은 과정으로 게임 세상의 이주가 진행된다.

부하 평형 조건이 되면, 전체 시스템을 관리하는 메인 서버는 이주할 게임 세상 프로세스(구 게임 세상 프로세스)에게 이주 준비 패킷을 전송하고, 그 패킷은 해당 게임 세상에 접속된 모든 클라이언트에게 전달된다. 이주 준비 패킷을 받은 클라이언트는 기존의 접속을 유지한 채(게임은 계속 진행되는 중), 새로 이주할 서버의 게임 세상 프로세스(새 게임 세상 프로세스)에 접속한다. 새 게임 세상 프로세스에 모든 클라이언트의 접속이 완료되면, 해당 프로세스는 메인 서버로 그 사실을 통보한다. 그러면 메인 서버는 구 게임 세상으로 이주 개시 패킷을 전송한다. 다시 구 게임 세상은 게임 서비스를 중지하고 자신의 데이터를 데이터베이스에 모두 저장한다. 이와 동시에 클라이언트들에게 이주 개시 패킷을 전송한다. 그러면 클라이언트들은 이미 만들어진 새 게임 세상 프로세스와의 접속을 이용하여 게임 패킷을 전송한다. 한편, 저장이 완료된 구 게임 프로세스는 메인 서버로 완료 통보를 한다. 그러면 메인 서버는 새 게임 세상에 패킷을 전송하여 데이터베이스로부터 데이터를 읽어 들이도록 한다. 새 게임 세상은 클라이언트에서 전송한 패킷을 처리하면서 필요한 데이터를 데이터베이스에서 읽어 들인다. 최종적으로 구 게임 세상과의 접속을 종료하면 서버 프로세스의 이주가 모두 완료된다.

IV. 구 현

제안된 부하 평형 기법을 검증하기 위하여 간단한 게임 서버 시스템을 구현하였다. 그림 **2** 는 전체 게임 서버 시스템의 구조를 나타낸다. 시스템은 메인 서버와 다수의 게임 서버, 그리고 DB 서버로 이루어진다. 메인 서버는 전체 게임 서버를 관리하며, 로그인 및 모니터링 기능을 수행한다. 게임 서버들은 게임 서버 프로세스들을 생성하여 실제 게임을 진행시키는 역할을 한다. DB 서버는 사용자 및 게임에서 다루는 모든 물체의 데이터를 관리한다.

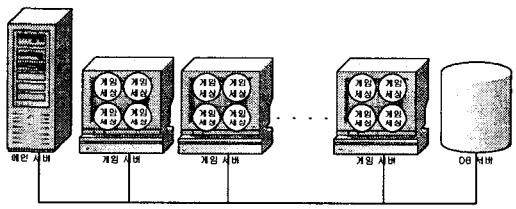


그림 2. 게임 서버 시스템 구조

4.1 게임 서버 프로세스의 구조

게임 서버 프로세스는 하나의 게임 세상을 담당하는 프로세스이다. 본 논문에서 구현된 게임 서버 프로세스는 메인 쓰레드 이외에 5 개의 쓰레드 구조로 구성되어 있다. 입력을 담당하는 입력 쓰레드, 게임에 대한 연산 처리 및 출력을 담당하는 게임 쓰레드, DB 를 관리하는 DB 쓰레드, NPC(Non-Player Character)의 행동을 처리하는 인공지능 쓰레드, 메인 서버와의 통신을 담당하는 모니터 쓰레드로 이루어져 있다.

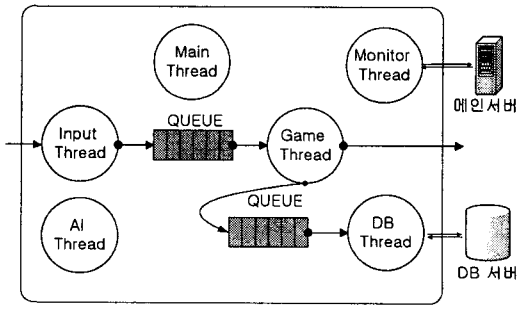


그림 3. 게임 서버 프로세스 구조

4.2 클라이언트 프로세스의 구조

본 논문에서는 구현된 서버를 테스트하기 위해서 두 종류의 클라이언트를 구현하였다. 첫째는 접속된 모든 사용자의 위치 및 정보를 표시해 주는 모니터 클라이언트이고, 둘째는 서버에 접속하여 데이터를 송수신하는 역할을 수행하는 더미 클라이언트이다.

모니터 클라이언트는 MS 윈도우 기반에서 동작하도록 제작한 프로그램으로서 전체 맵을 보여주고 더미 클라이언트의 ID 와 좌표 정보를 표시해 주는 프로그램이다. '사용자 모드'와 '모니터 모드' 두 가지 형태의 화면을 제공하며, 화면에 표시되는 정보에 차이가 있다. 사용자 모드에서는 ID 와 X 좌표, Y 좌표를 보여주고, 모니터 모드에서는 전체 맵에 있는 자기 자신과 다른 더미 클라이언트의 위치를 점으로 표시한다.

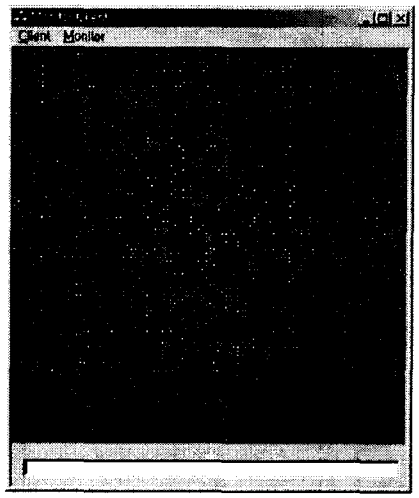


그림 4. 모니터 클라이언트(모니터 모드)

더미 클라이언트는 부하 테스트를 위해 FreeBSD 기반에서 동작하도록 제작한 프로그램이며 실제 게임 환경에서는 MS 윈도우 기반에서 실행되는 하나의 게임 클라이언트의 역할을 하는 프로그램이다. 더미 클라이언트에서 ID 와 좌표정보를 저장한 패킷을 서버로 전송하면 다시 서버로부터 같은 내용의 패킷을 수신하도록 하였다. 하나의 더미 클라이언트는 일정한 시간 간격으로 패킷을 전송하도록 하였으며, 테스트 환경의 제약으로 인해 수백 개의 더미 클라이언트를 동시에 실행시켜 서버에 접속하여 패킷을 송수신 하도록 하였다.

실험에서 한 개의 클라이언트는 초당 하나의 패킷을 전송하도록 하였고, 패킷 하나의 크기는 28~34 바이트로 정하였다. 그림 1 과 같은 상황에서 실험한 결과, 게임 세상의 이주가 성공적으로 수행되었다. 그리고 500 개의 클라이언트와 새 게임 세상과의 접속을 미리 만드는 데에 38 초의 시간이 소요되었다. 만약 제안된 바와 같이 미리 새 게임 서버로 접속하는 과정이 없다면 이 시간은 모두 게임 서비스의 지연 시간에 반영될 요소일 것이다. 현재 데이터베이스 서비스를 담당할 DB 서버가 개발 중이다. 게임 세상 이주 시의 실제적인 서비스 지연은 DB 서버 개발 이후에 실험할 예정이다.

V. 결 론

본 논문에서는 지역 서버 구조를 갖는 온라인 게임에서 온라인으로 서버의 부하를 분산시키는 기법을 제안하였다. 제안된 기법에서는 필요에 따라 특정 게임 세상을 운용하는 게임 서버를 다른 게임 서버로 전환시켜 게임 세상을 이주시켰다. 이 과정에서 서비스의 중단 시간을 줄이기 위해 게임을 진행하면서 새로운 게임 서버로의 접속을 미리 만들도록 하였다. 또한 부하를 발생시킬 수 있는 가상의 클라이언트를 제작하여 제안된 기법을 검증하였다. 실험 결과 프로세서의 이주가 진행되는 동안에 클라이언트들의 서비스가 원활히 유지되면서 부하 평형이 완료됨을 확인하였다.

제안된 부하 평형 기법의 성능을 파악하기 위해서 보다 실질적인 테스트 환경에서의 테스트가 필요하다. 또한, 제안된 기법의 접속 및 데이터 송수신 처리를 담당하는 커넥션 서버의 도입에 대한 연구도 진행 중이다. 제안된 기법은 실제의 온라인 게임에 적용시킬 예정이다.

참고 문헌

[1] 황요한, 김동균, 장인걸, 신동일, 김동현, "Multimedia Storage Server 를 응용한 MMO 게임 서버 구조에 대한 설계," 한국정보과학회 2002 년 추계학술발표논문집(2), 제 29 권 2 호, pp. 211~213, 2002.

[2] 김상균, "대규모 네트웍 가상환경을 위한 동적 로드 밸런싱," 한국과학기술원 석사 학위논문, 2000.

[3] 이경민, 이동만, 김성훈, 장소희, 박은광, 현순주, "대규모 다중 서버 분산 가상 환경 시스템을 위한 확장성 있는 로드 밸런싱 기법," 한국정보과학회 2002 년 추계학술발표논문집(3), 제 29 권 2 호, pp. 160~162, 2002.