

MMORPG 부하 분산을 위한 동적 맵 분할 시스템 설계

이정진⁰ 두길수 안동연 정성중
전북대학교

{jeongjin2⁰, dgs, duan, sjchung}@moak.chonbuk.ac.kr

Design of Dynamic Map-Divide System for Load Distribution of MMORPG (Massively Multi-player Online Role Playing Game)

Jeongjin Lee⁰, Gilsoo Doo, Dongun Ann, Seungjong Chung
Dept. of Computer Engineering, Chonbuk National University

요약

오늘날의 대규모 온라인 네트워크 게임은 동시에 수만에서 수십만 명이 접속하여 게임을 즐기는 서버/클라이언트 개념을 사용하고 있다. 접속하는 클라이언트의 수가 증가함에 따라 서버에서 다수의 클라이언트 유저를 지원하기 위해서는 게임 서버가 이를 처리할 수 있어야 한다. 이를 위해 게임서버 구조에 대한 연구들이 이루어지고 있다. 하지만 현재까지의 많은 서버들이 유동적인 클라이언트를 정적인 시스템으로 대처하기엔 문제점을 가지고 있다. 본 논문에서는 기존 게임서버에서 부하 분산방식의 구조적 단점을 보완하고, 부하를 최소화 할 수 있는 동적 부하 분산 방법을 제안한다.

1. 서론

세계적으로 네트워크의 발전과 함께 온라인 게임도 급속도로 확산 보급되었으며, 또한 고부가가치를 창출하는 미래의 핵심 산업으로 각광을 받고 있다. 이런 온라인 게임은 최대한 많은 사용자를 수용하고, 사용자들에게 양질의 서비스를 안정적으로 제공하는 서버의 역할이 가장 중요하다고 볼 수 있다. 이를 위해서는 다수의 사용자들이 하나의 서버에 집중되는 구조를 피하고, 과 부하된 서버의 부하를 분산시키는 방법이 필요하며 이에 대한 연구가 활발하게 진행되어지고 있다.

본 논문에서는 다수의 사용자가 접속 가능한 MMORPG (Massively Multi-player Online Role Playing Game) 환경 하에서 동적인 부하 분산 방법을 제시하였다.

현재 온라인 게임서버의 부하 분산 방법은 데이터를 분산시키는 맵 분할 방식, 프로세싱을 분산시키는 클러스터링 부하 분산 방식, VDG^[1] (Virtual Data Grouping, 데이터 가상 그룹핑) 방법 등이 복합적으로 이용되고 있다. 하지만, 현재의 분산 시스템은 유동적인 사용자에 의한 게임서버 한곳으로의 집중 현상에 대해 매우 정적인 시스템 구조를 가지고 있다는 단점이 있다.

그래서 본 논문에서는 그 해결 방안으로 맵을 Cell과 Block단위로 나뉘 이것을 기반으로 시스템을 실시간으로 동적 분산하는 모델을 제안한다.

본문에서는 최근 연구되는 부하 분산 관련 연구들을 살펴보고 문제점들을 지적 하며, 동적 부하 분산의 이론과 시스템 구조에 대해 제안할 것이다.

2. 관련 연구

2.1 연구 사례

2.1.1 Optimal Grid^[2]

Optimal Grid는 Grid 기술을 적용하여, FEM (Finite Element Modeling, 유한요소모델링)을 기반으로 어떠한 문제에 대해 이웃간의 연결 상관관계를 정의할 수 있는 요소들로 분리하여 문제를 해결하는 방식이다. IBM Almaden Research Lab에서 연구 개발하였고, Grid 화를 위한 통합 환경과 문제 해결 환경을 모두 포함하는 미들웨어 기반 연구이다.

2.1.2 CORBA 기반 분산 컴퓨팅^[3]

이종의 분산된 환경 하에서 응용 프로그램들을 서로 통합할 수 있도록 CORBA를 사용하였고, CORBA 자체적으로 제공하는 IIOP를 이용하고, 동적인 쓰레드 생성 및 다른 서버로의 connection을 해중으로써 좀더 효율적인 동시성 제어 및 네트워크 부하를 줄일 수 있도록 제안되었다.

2.1.3 Butterfly Grid^[4]

과학 연산 분야에서만 쓰이던 Grid 기술을 게임 네트워크에 적용할 수 있는 시스템 구조로 설계하였다. 오픈 소스인 미들웨어 Globus Toolkit^[5]에 기반으로 자체 개발한 시스템으로 구축하였으며, 사용자를 상황에 따라 서버를 넘나들 수 있도록 제안된 시스템이다.

2.2 기존 기술의 문제점 및 한계

본 논문에서 제안하고자 하는 분산 방법은 데이터를 분산하는 방법인 맵 분산방식을 모태로 한다. 기존의 맵 분산 방식은 지정된 영역별로 개별 게임서버가 프로세싱

하도록 설계 되어 있어 시스템이 부하에 관하여 매우 정적이다. 이러한 경우 유동적인 사용자들에 의해 실시간으로 어느 한곳이 몰리는 현상이 발생할 것이고, 심한 경우 시스템이 다운되는 우려가 현실로 발생할 것이다.

프로세싱 분산 방식인 VDG 방식을 통해서 클러스터링하는 방법 또한 앞의 문제에 대해선 장점을 지니지만, 최대 받아들일 수 있는 사용자 Maximum은 존재하고 그 한계를 넘을 수는 없다. 온라인 게임에서 하나의 매력인 많은 사용자와의 커뮤니티를 포기하면서 시스템을 구축할 수는 없을 것이다.

이러한 문제점을 해결하려, 최근 동적으로 시스템을 분배하는 분야가 연구 중에 있다. 그중 OptimalGrid 기술의 미들웨어를 이용한 시스템은 오픈 소스가 아니고 서버와 통합된 환경이기 때문에 시스템을 구축하기 위해서는 고가의 비용을 들여야 하며 그만큼 응용 할 수 있는 영역이 한정적이 될 수밖에 없을 것이다.

CORBA나 Globus Toolkit, 등을 이용한 다른 연구 분야들은 MMORPG 게임 환경에 한계가 있으며, 단지 일정 수준의 유저를 위한 게임서버환경에서 동적으로 응용할 수 있는 시스템만을 구축할 수 있는 분야 들이다.

그래서 본 논문에서는 아래 표 1과 같이 기존 맵 분산 방식의 유저 집중현상을 피하고, OptimalGrid처럼 동적인 시스템이지만, 구축비용이 적게 들고, 통합 환경이 아닌 응용성이 좋은 시스템을 제안하는 것이다.

Computation Complexity는 OptimalGrid가 미들웨어 단에서 작은 CELL단위로 분할하는 것이므로 가장 높고, VDG방식 클러스터링, 기타 방식들은 게임 서버의 응용 프로그램에 알고리즘을 적용, 구현 제작하는 것이므로 낮다. 본 논문에서 제안한 시스템은 게임서버에 Piece Cell 단위로 분할 알고리즘을 적용한 것이므로 OptimalGrid보다는 낮지만, 기존 시스템보다는 높다.

3. 제안된 동적 분산 시스템 이론과 구조

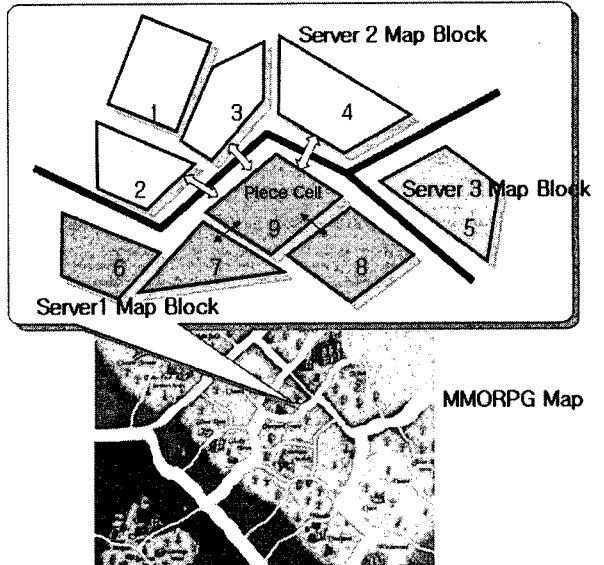
3.1 동적 분산 이론

기본 개념은 Grid 개념을 도입하였다.^[2] 즉, 하나의 큰 문제를 작은 조각들로 나누어 생각하자는 것이다. 그리고 각 서버마다 성능에 따라 몇 개의 작은 조각들을 처리하도록 하고, 부하가 심해지면, 인근의 유휴자원을 활용하여 부하가 심한만큼의 작은 조각들을 다른 서버에서 처리하도록 하자는 것이다.

실제 시스템에선, 다수의 사용자가 이용할 큰 맵을 활동의 영역에 따라 다수의 조각 셀 (Piece Cell)로 나누고, 게임서버에서는 처리할 수 있는 능력에 따라 다수개의 Piece Cell을 담당하고, 그 맵 위에서 활동하는 사용자만큼을 담당하여 프로세싱하자는 것이다.

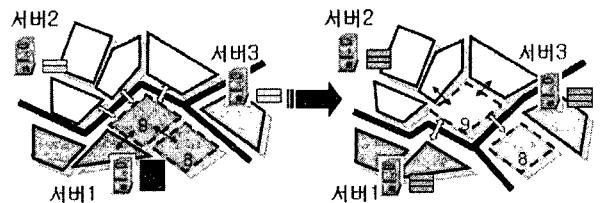
한 서버가 담당할 수 있는 Piece Cell들을 Map Block라 하고, 그 안에 Piece Cell들은 서로 독립되게 실행하

게 된다. Map Block안의 인접한 Piece Cell들 간은 내부 통신 연결이 이루어지고 인접한 Piece Cell이 다른 Map Block 안에 있다면, 네트워크를 통해서 인접 서버로 데이터를 전송하게 된다.



(그림 1) Map Block 내의 서로 나뉜 Piece Cell

그림1에서 Piece Cell 9는 같은 Map Block안의 6, 7, 8 Piece Cell과 내부 통신을 하게 되고, 인접하지만 다른 Map Block인 2, 3, 4 Piece Cell과는 외부 통신을 하게 된다. 만약 Piece Cell 7로 사용자가 집중 될 경우, 서버1은 부하가 심해지게 된다. 이러한 경우 Piece Cell 9를 서버2의 Map Block으로 보내고, 서버1의 Map Block은 담당하는 Piece Cell의 수를 줄여서 부하를 낮추게 한다.



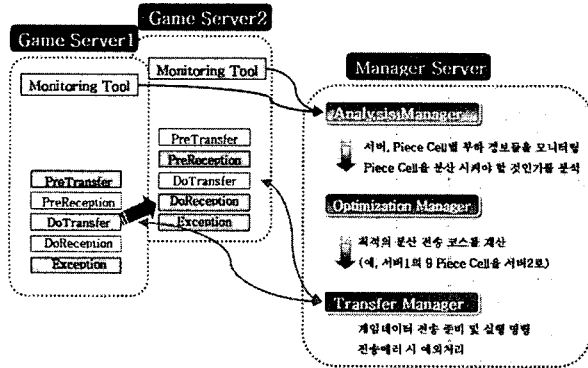
(그림 2) 서버1의 부하가 심해질 경우, 그 Map Block 안의 적정개의 Piece Cell들을 서버2,3에 분산시킴

그림 2와 같이 실시간으로 부하가 어느 한계점을 넘으면 자동으로 인접 Map Block중 부하가 낮은 서버를 검색하고, 부하 균형을 서버들 간 맞추기 위해서, 몇 개의 Piece Cell을 순간적으로 이동시켜서 실행하게 만드는 것이다. 그림 2에서는 Piece Cell 8, 9가 서버1 Map Block에서 각각 서버3, 서버2로 이동하였다. 물론 Piece Cell의 모든 환경을 옮기며, 서버가 옮겨졌다는 것을 사용자가 알 수 없도록 해야 할 것이다.

3.2 시스템의 구조

그림 3은 본 논문에서 설계한 동적 분산 시스템의 전체적인 구조를 나타낸 것이다.

기존의 게임 시스템에서 Manager Server를 새로이 추가했다. Manager Server의 주요 기능은 부하 측정 데이터로 최적의 계산을 하고, 최적의 분산 전송 코스를 계산하며, Game Server들에 명령을 하달하여 이동 준비 및 전송을 하는 것이다.



(그림 3) 동적 분산 시스템의 구조

산하는 역할을 한다.

Transfer Manager에서는 Game Server의 PreTransfer, PreReception, DoTransfer, DoReception을 이용하여 게임데이터 전송 준비 및 이동 명령을 내리며, 실행 중에 예러가 발생하면 이전상태로 데이터를 복귀시키거나, 네트워크 오류 시 데이터를 재전송하는 Exception 기능이 있다.

4. 결론 및 향후 연구 방향

제안된 시스템은 MMORPG 환경 하에서 동적으로 부하를 분산 하도록 설계이다. 이를 운용함으로써 게임 중 하나의 서버에 집중 되는 부하를 인근의 사용이 적은 서버에 분산 시켜줌으로써 문제를 해결한다.

그래서 기존 게임 시스템 구조에서 단지 Manager Server만이 새로이 추가한 것이고, 미들웨어와 연동하여 제작한 것이 아니어서 쉽게 이용, 응용이 가능하다. 즉 기존의 부하 분산 방식과 연동도 가능하며, 게임이 아닌 다른 분야에도 제안된 시스템을 적용할 수 있다.

앞으로 연구방향은 상황에 따라 Piece Cell을 어떻게 병합할 것인가에 대한 알고리즘 연구와 제안된 시스템에서 Monitoring Tool과, Manager 서버를 통해 맵을 분할, 제어 하는 과정을 구현할 예정이다.

참 고 문 헌

- [1] 이철민, 박홍성. "다중 사용자 게임 성능 향상을 위한 데이터 가상 그룹핑 방법", 『정보과학회논문지』 소프트웨어 및 응용 제 30권 제3호, pp.231-238, 2003.04
- [2] Tobin J, Lehman, James H. Kaufman. "OptimalGrid: Middleware for Automatic Deployment of Distributed FEM Problems on an Internet-Based Computing Grid", 『IEEE International Conference on Cluster Computing(CLUSTR'03)』, 2003.12
- [3] 최재언, 이해원, 하수철. "CORBA 기반 분산 네트워크 게임 서버에 관한 연구" 『한국정보처리학회 2001년 춘계학술대회』, 제8권 제1호, pp.159-162, 2001.04
- [4] IBM Grid Research Lab "Butterfly.net: Powering Next-Generation Gaming with Computing On-Demand" 『Butterfly.net』 <http://www.butterfly.net>
- [5] 김동균 "그리드 컴퓨팅(기반의 네트워크 게임서버 설계)" 『한국정보과학회 학회지』 제20권 제2호 pp.5~10 2002.02
- [6] 이남재, 박홍성 "온라인 RPG의 시리즈 시나리오(캠페인)를 위한 분산형 게임 서버 적용 방법" 『한국게임학회』 2002.01
- [7] 윤영미, 전재우, 오삼권 "대규모 네트워크 게임을 위한 게임 서버 구조" 『한국정보처리학회 2002년 추계학술대회』 제9권 제2호 2002.10
- [8] 황요환 외 4명. "Multimedia Storage Server를 응용한 MMO 게임 서버 구조에 대한 설계", 『한국정보과학회 2002년 추계학술대회』 제29권 제2호, 2002.10
- [9] <http://www.ggf.org>

또한, 게임 서버에서는 기존 서버에서 부하 량을 측정하는 Monitoring Tool과, 동적 분산에 필요한 PreTransfer, PreReception, DoTransfer, DoReception, Exception이 추가되었다.

부하 량 계산은 Game Server의 Monitoring Tool에 의해서 측정된 Usage Record를 가지고 계산한다. 본 논문에서 제안한 시스템의 Usage Record의 Minimum Set은 표 1과 같다.

특징적인 부분은 단지 유저의 수만으로 부하 량을 측정한다는 것이다, 단 맵 지형별 유저의 평균적인 활동성을 체크하여 Piece Cell별로 적용하였다.

Num	Field
1	Game Server Name
2	Piece Cell Number
3	Activity of User
4	The number of User

(표 1) Usage Record의 Minimum Set

3.3 시스템의 기능

기존의 게임서버에서 추가된 Monitoring Tool은 매 분마다 리소스를 체크하여 Analysis Manager로 부하 량 데이터를 전송한다.

Manager Server의 Analysis Manager에서는 전송된 Usage Record Set을 가지고 각 서버별, Piece Cell별 부하량을 산출하여 맵 분할이 필요한지 여부를 계산한다.

Optimization Manager는 최적의 분산 전송 코스를 계