다중 사용자 게임 성능 향상을 위한 데이터 그룹핑 방법

이철민, 박홍성
강원대학교 제어계측공학과

Data Grouping for Performance Enhancement of Multi-User Games

Chul-Min Lee, Hong-Seong Park
Dept. of Control Instrument Engineering, Kangwon Nat' l University
E-mail : jazonsim@control.kangwon.ac.kr, hspark@cc.kangwon.ac.kr

요 약

현재 많은 사용자들이 네트워크 게임을 즐기고 있다. 네트워크 게임은 동시에 많은 사용자들이 하나의 서버에 접속하여 게임을 하므로, 많은 사용자들이 접속하는 경우 서버의 부하 증가와 네트워크 트래픽 증가로 인한 데이터 전송 지연이나 데이터 손실이 발생하여 게임을 원활히 진행하지 못하는 문제가 있다. 본 논문에서는 이러한 지연을 줄이기 위해서 그룹화 방법을 이용하여 서버가 전송하는 데이터의 개수를 줄임으로써 서버의 부하 감소 및 네트워크 트래픽을 줄이는 방법을 제안한다.

1. 서론

인터넷이 발전함에 따라 게임 산업에도 많은 변화가 일어났다. 기존의 게임은 사용자 혼자서 즐기는 싱글 게임이나 모뎀을 이용하여 4 ~ 8명 정도의 사용자들이 같이 게임을 즐기는 정도의 게임이 주류를 이루고 있었다. 그러나 현재는 인터넷을 기반으로 하여 3000 ~ 4000명이 동시에 하나의 서버에 접속하여 게임을 즐기는 네트워크 게임이 주류로 발전하고 있다.

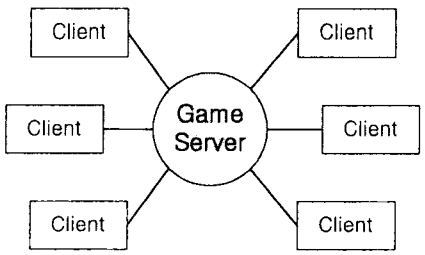

그림 1 네트워크 게임의 서버/클라이언트 구조

일반적인 네트워크 게임의 구조는 그림 1처럼 하나의 서버와 다수의 클라이언트들로 구성된다. 서버는 클라이언트들로부터 수신되는 요청을 처리하여 그 결과를 클라이언트에 전송하는 역할을 하고, 클라이언트는 서버로부터 수신한 응답 결과에 따라 게임을 진행하게 된다.

네트워크 게임은 하나의 서버에 다수의 클라이언트들이 접속해서 동시에 게임을 진행하므로 같이 게임을 하고 있는 다른 클라이언트들의 현재 상태나 위치, 동작 등에 대한 정보가 매우 중요하다. 만약 이러한 정보들이 동기화 되지 않는다면, 클라이언트마다 게임 상황이 다르게 되어 정상적인 게임을 진행할 수 없게 된다. 이러한 문제를 해결하기 위해서는 주기적으로 각 클라이언트마다 다른 클라이언트들의 정보를 전송해주어야 하는데, 만약 N 명의 사용자들이 게임을 하고 있다면, 서버가 전송해야하는 데이터 개수는 $N \times N$ 이 된다. 그렇기 때문에 접속하는 클라이언트의 수가 증가할 수 록 클라이언트의 요청을 처리하는 게임 서버와 데이터가 송수신 되는 네트워크에는 더욱 많은 부하가 걸리게 된다. 이렇게 서버와 네트워크에 많은 부하가 걸리게 되는 경우, 클라이언트는 서버의 처리 지연이나 네트워크 트래픽 증가로 인한 네트워크 전송 지연 때문에 해당 요청에 대한 응답을 늦게 수신하거나, 최악의 경우 과도한 네트워크 트래픽으로 인한 패킷 손실이 발생하여 응답을 받지 못하는 경우가 존재한다.

응답 결과를 다른 사용자보다 늦게 받는다면 즉 응답이 지연되면 해당 처리가 다른 사용자보다 늦기 때문에 게임 결과가 달라지고, 패킷 손실로 서버로부터 응답을 받지 못하는 경우, 즉 응답 손실의 경우 사용자는 해당하는 처리를 못하게 되어 게임을 진행하지 못하게 된다. 따라서 네트워크 게임을 원활하게 운영하기 위해서는 이러한 지연이나 손실 문제들을 해결할 필요가 있다.

기존의 연구[1][2]에서는 네트워크 게임의 성능을 평가하였다. [1]에서는 서버 없이 클라이언트들이 $N\times N$ 연결을 이루는 경우의 동기화 성능을 분석하여 동기화가 유지됨을 보여줬으나, 적은 수의 클라이언트들이 연결된 경우를 대상으로 한 성능분석이고 전송회수를 줄일 수 있는 방법이 없기 때문에 많은 클라이언트가 연결되는 경우 클라이언트의 부하와 네트워크의 부하가 심하게 걸려 동기화 유지가 불가능하다. [2]에서는 LAN에서 아주 빠른 데이터 송수신이 일어나는 경우의 성능을 분석하여 성능에 영향을 미치는 요인을 클라이언트의 CPU 성능이라 분석하였으나, 성능 평가 환경이 로컬 네트워크에서 적은 수의 클라이언트들이 서버에 연결된 경우를 대상으로 하였으므로, 다수의 클라이언트가 연결되어 전송 횟수가 증가하고 네트워크 트래픽이 증가하는 경우 실제 성능에 영향을 미치는 요인은 서버의 처리 지연과 네트워크에서의 지연이 된다.

네트워크 게임에서의 지연을 해결하기 위해 여러 방법이 제안되었는데, 네트워크로 전송되는 패킷 크기를 줄이거나, 프로그램 구조개선, 지연을 감추는 기법 등을 통해 지연을 줄이는 방법[3]과 서버 프로그램의 구조 개선이나 분산 서버 구조 등을 사용하여 서버의 지연을 최소화하는 방법[4], 그리고 서버의 사용자들을 그룹화 하여 서버가 전송해야할 데이터의 수를 줄여 네트워크 부하 및 서버의 부하를 줄여 지연을 최소화하는 방법[5]등이 있다

[3]에서 제안하는 방법을 사용하는 경우 지연을 줄이거나 감출 수는 있지만, 패킷의 크기를 줄이는 것은 한계가 있고, 패킷의 크기를 줄이기 위해 압축방법을 사용하는 경우 서버와 클라이언트 모두에게 처리부하를 증가시키므로 성능 향상에는 한계가 있을 수밖에 없다. 또한 전송량을 줄이는 방안이 없으므로 클라이언트 수가 증가하는 경우 전송 개수는 $N\times N$ 로 증가하는 문제가 있다. [4]에서 제안하는 방법을 사용하는 경우, 서버 프로그램 구조 개선은 개발하는 응용에 따라 많은 변화가 있는 부분이므로 모든 응용에 적용하기 어렵고, 분산 서버 구조를 사용하면 서버의 부하를 분산하여 서버 지연을 해결하는데 도움이 되나, 추가적인 하드웨어 등이 필요하므로 비용이 증가하는 등 분산 서버의 제어에 문제가 존재한다. [5]에서는 클라이언트 수가 증가함에 따라 서버의 전송회수가 기하급수적으로 증가하는 문제를 해결하기 위해, 맵을 특정 크기의 영역으로 분할하고 그 영역에 있는 클라이언트들을 하나의 그룹으로 묶어, 그룹 단위로 클라이언트 상태를 전송하도록 하여 서버의 데이터 전송 횟수를 줄일 수 있는 방법을 제안하였다. 그룹 단위 전송 방법을 사용하는 경우 데이터 전송 개수가 줄어드는 장점이 있으나, 서버의 전송 횟수를 더욱 줄이기 위해서는 영역을 더욱 작게 나누어야하므로 영역을 관리하는 처리 프로세스 개수가 증가하게 되어 서버의 부하가 증가하는 문제가 있었다.

본 논문에서는 기존 그룹 단위 전송 방법[5]를 개선하여 그룹에 포함된 클라이언트들을 가상 그룹으로 분할하는 방법을 사용하여 영역을 더 작게 나누지 않고도 전송 횟수를 줄여 서버의 부하와 네트워크 부하를 동시에 줄일 수 있는 방법을 제안하고, 성능 분석을 통해 기존의 방법[5]와 비교하여 이 방법의 유효성을 보이겠다. 2장에서는 기존 그룹 단위 전송 방법[5]에 대해서 기술하고, 3장에서는 가상 그룹화 방법을 소개하고, 4장에서는 그룹 단위 전송 방법과 가상 그룹화 방법의 성능 분석 및 비교한 후 5장에서 결론을 맺겠다.

2. 그룹 단위 전송 방법

네트워크 게임은 여러 사용자들이 함께 게임을 하므로 계속적으로 다른 사용자들의 정보나 동작 등에 대한 정보가 필요하다. 만약 이러한 정보를 클라이언트마다 모두 전송한다면, $N\times N$ 개수의 전송이 일어나므로, 맵을 특정 크기의 영역으로 그룹화 하여 해당 그룹에게만 데이터를 전송한다면, 전송 횟수도 줄이고, 처리 부하 또한 줄 일 수 있게 된다. 다음 그림 2는 전체 맵을 25개의 영역으로 분할 한 예이다.

1	2	3	4	5
6	7	8	9	10
11	12	12	14	15
16	17	18	19	20
21	22	23	24	25

그림 2. 25개로 분할된 맵

본 논문에서는 다음과 같이 가정한다.

1. 클라이언트는 맵에 고르게 분포한다.
2. 맵은 Z 개의 크기가 같은 영역으로 분할되고, 각 영역에는 동일한 수 (전체 클라이언트수 수/Z)의 클라이언트들이 있다.
3. 같은 영역의 클라이언트들은 하나의 그룹이 된다.
4. 각 그룹마다 별도의 프로세스가 관리한다.
5. 최악의 경우, 클라이언트는 최소 전송 주기 T_S 마다 서버로 K byte의 프레임을 전송한다.
6. 서버는 클라이언트가 전송한 프레임을 수신한 후 T_S 시간 내에, 처리 결과를 이 프레임을 전송한 클라이언트 및 이 클라이언트가 속한 그룹의 클라이언트들에게 K byte의 프레임으로 전송한다.

또한 사용되는 기호는 다음과 같이 정의한다.

P_T : 게임상의 전체 플레이어 수

Z : 영역의 개수

P_Z : 특정 영역에 있는 플레이어 수

T_S : 클라이언트가 서버로 데이터를 전송하는 최소 시간 간격.

S_R : 최악의 경우, 서버가 T_S 마다 클라이언트들로부터 수신하게 되는 최대 프레임 개수

S_S : 최악의 경우, 서버가 T_S 마다 전송해야하는 최대 프레임 개수

S_{GR} : 최악의 경우, 한 그룹 프로세스가 T_S 마다 한 그룹으로부터 수신하게 되는 최대 프레임 개수

S_{GS} : 최악의 경우, 한 그룹 프로세스가 T_S 마다 그룹에게 전송해야하는 최대 프레임 개수

C_R : 최악의 경우, 클라이언트가 T_S 마다 수신하게 되는 최대 프레임 개수

그룹화 방법을 사용하지 않은 경우와 그룹화를 사용하는 경우를 비교해 보겠다. 먼저, 그룹화 방법을 사용하지 않고, 한 클라이언트가 서버로 보낸 프레임을 게임 상의 모든 클라이언트에게 데이터를 전송하는 경우를 살펴보면, 최악의 경우, T_S마다 서버가 수신하게 되는 최대 프레임 개수는 다음과 같다.

$$S_R = P_T * 1$$

최악의 경우, 서버가 T_S 마다 클라이언트들에게 전달해야 하는 최대 프레임 개수는 다음과 같다.

$$S_S = S_R * P_T$$

최악의 경우, 각각의 클라이언트들이 T_S 마다 수신하게 되는 최대 프레임 개수는 다음과 같다.

$$C_R = S_S / P_T = S_R$$

그룹화 방법을 사용하여, 해당 그룹에게만 데이터를 전송하는 경우를 살펴보면 다음과 같다.

최악의 경우, T_S 마다 서버가 수신하게 되는 최대 프레임 개수는 다음과 같다.

$$S_R = P_T * 1$$

최악의 경우, 그룹 프로세스가 T_S 마다 한 그룹으로부터 수신하게 되는 최대 프레임 개수와 전송해야 하는 최대 프레임 개수는 다음과 같다.

$$S_{GR} = P_Z * 1 \quad S_{GS} = S_{GR}$$

이 경우, 서버가 T_S 마다 클라이언트들에게 전달해야 하는 최대 프레임 개수는 다음과 같다.

$$S_S = S_{GR} * S_{GS} * Z$$

최악의 경우, 각각의 클라이언트들이 T_S 마다 수신하게 되는 최대 프레임 개수는 다음과 같다.

$$C_R = S_{GS}$$

$P_T = 10,000$일 때, 그룹화를 사용하지 않는 경우와 그룹화($Z = 100$ 사용)를 사용하는 경우, 최악의 경우 서버가 전송하게 되는 최대 전송 개수와 클라이언트의 최대 수신 개수를 비교해 보면 다음 표1과 같다.

표 1 모든 클라이언트에 전송하는 경우와 그룹화 비교

	그룹화 사용 안 함	그룹화 사용
서버 전송 개수	100,000,000	1,000,000
클라이언트 수신 개수	10,000	100

위의 비교를 살펴보면, 맵을 100개로 분할한 경우, 자신의 그룹에게만 전송하는 경우보다 1/100로 서버의 전송 개수 줄어들게 된다.

그룹화 방법을 사용하는 것이 그렇지 않은 것 보다 효율적이지만 그룹화한 경우 다음과 같은 문제점이 있다. 다음 그림 3을 보면 그룹 12에 있는 클라이언트 A는 그룹 1, 2, 11에 인접하여 있기 때문에 실제 필요한 데이터는 그룹 1, 2, 11, 12의 데이터가 된다. 만약 자신의 그룹의 데이터만 수신하게 되면 실제 필요한 데이터를 수신하지 못하는 문제점이 있다.

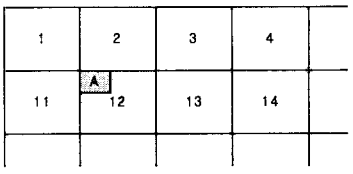

그림 3. 주변 그룹과 인접한 클라이언트

만약, 현재의 그룹과 주변의 그룹을 포함한 데이터 전송 그룹을 만들어 전송 그룹 전체에 데이터를 전송하도록 하면 이러한 문제를 해결할 수 있다. 이러한

방법을 그룹 단위 전송 방법이라 부르겠다.

자신의 그룹을 포함하여 주변 그룹에까지 데이터를 전송하는 경우, 자신의 그룹과 주변 그룹에 전송하는 비율에 차등을 둬서 전송하게 되면 전송 횟수를 줄일 수 있다. 예로, 자신의 그룹에는 T_S 마다 전송하고 주변 그룹에는 10 T_S 마다 전송하는 등의 차등을 두면 서버의 데이터 전송 횟수를 줄일 수 있다.

한 그룹에서 주변 그룹에 데이터를 전송하는 경우에는 3가지 경우가 있는데, 다음 그림 4를 참고하면, 자신의 그룹 주위에 8개의 그룹이 있는 경우(예, 그룹 25)와 5개의 그룹이 있는 경우 (예 그룹 30), 3개의 그룹이 있는 경우(예, 그룹1)가 있다.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	18	19	50

그림 4. 경우에 따른 주변 그룹수

맵을 Z개의 영역으로 분할할 때 현 그룹 주위에 몇 개의 그룹이 있는가는 다음과 같다.

8개의 그룹이 존재하는 경우의 수 : $(\sqrt{Z}-2)^2$

5개의 그룹이 존재하는 경우의 수 : $(\sqrt{Z}-2)*4$

3개의 그룹이 존재하는 경우의 수 : 4

현 그룹과 주변 그룹의 전송 비율을 R 이라고 할 때, 최악의 경우, 서버가 T_S 마다 클라이언트들에게 전달해야 되는 최대 프레임 개수는 다음과 같다.

$$\begin{aligned} S_S = & S_{GR}(P_Z+8RP_Z)(\sqrt{Z}-2)^2 \\ & + S_{GR}(P_Z+5RP_Z)(\sqrt{Z}-2)*4 \\ & + S_{GR}(P_Z+3RP_Z)*4 \\ = & S_{GR}*P_Z\{(8R+1)*Z-12R\sqrt{Z}+4R\} \end{aligned}$$

이 경우 클라이언트가 T_S 마다 수신하게 되는 평균 프레임 개수는 다음과 같다.

$$C_R = \frac{S_S}{P_T} = \frac{S_{GR}\{(8R+1)*Z-12R\sqrt{Z}+4R\}}{Z}$$

3. 가상 그룹화

현 그룹에서 발생한 데이터를 주변 그룹에 전송할 때 비율 R로 전송하는 경우에, 주변 그룹에 정확한 정보를 전송하기 위해서 비율 R을 1에 가깝게 하면, 전송할 데이터 개수 증가하고, 만약, 비율 R을 아주 작게 줄이면 주변 그룹에 정보를 전송하는 의미가 없어지게 된다.

또한 다음 그림 5와 같이 그룹이 이루어진 경우, 그룹 25를 가상 영역 25a, 25b, 25c, 25d로 구분했을 때, 25a 영역에 있는 클라이언트의 데이터는 그룹 14, 15, 24에게 의미가 있지 수 있지만, 그룹 16, 26, 34, 35, 36에게는 큰 의미가 없게 된다.

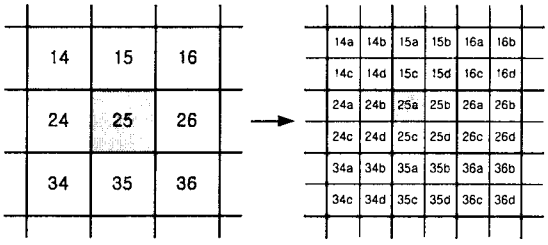

그림 5. 그룹을 4개의 가상 영역으로 분할

본 논문에서 이러한 경우를 고려하여, 한 그룹을 4개의 가상의 영역으로 나누어, 각 그룹을 관리하는 프로세스가 데이터가 어떤 가상영역에서 발생했는가를 주변 그룹에 알려줘 주변 그룹은 자신의 가상 영역 중 적당한 영역에 있는 클라이언트들에게 전송하도록 하고, 최신의 정보를 전달하기 위해 전송 비율 R은 1을 사용하는 가상 그룹화 방법을 제안한다.

한 그룹에서 주변 그룹의 일부 클라이언트에게 데이터를 전송하는 경우에는 3가지 경우가 있다. 먼저 주변에 8개의 그룹이 있는 경우, 데이터를 수신하게 되는 주변 그룹의 가상 영역 개수는 5개가된다. (영역으로 환산하면 1.25 영역)

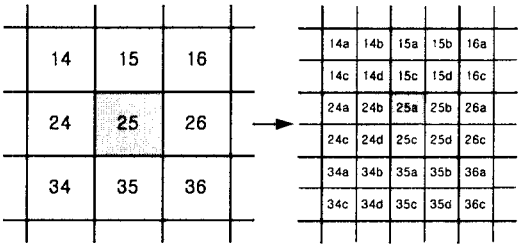

그림 6 주변에 8개의 그룹이 있는 경우

주변에 5개의 그룹이 있는 경우, 데이터를 수신하게 되는 주변 그룹의 가상 영역 개수는 (2+5+5+2)/4개가 된다. (영역으로 환산하면 14/16 영역)

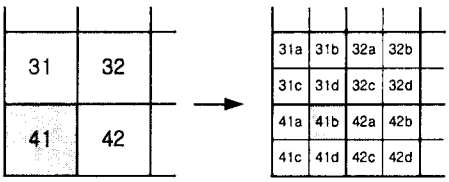

그림 7 주변에 5개의 그룹이 있는 경우

주변에 3개의 그룹이 있는 경우, 데이터를 수신하게 되는 주변 그룹의 가상 영역 개수는 (2+5+2)/3 이 된다. (영역으로 환산하면 9/12 영역)

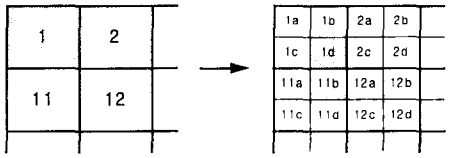

그림 8. 주변에 3개의 그룹이 있는 경우

최악의 경우, 클라이언트가 서버로 데이터를 보내는 최소 주기 T_S 동안, 그룹 프로세스가 한 그룹으로부터 수신하게 되는 최대의 프레임 개수는 다음과 같다.

$$S_{GR} = P_Z * 1$$

이 경우, 주기 T_S 동안 서버가 전송해야하는 최대 프레임 개수는 다음과 같다.

$$S_S = S_{GR} * P_Z\{(1.25R+1)Z - 1.5R\sqrt{Z} + R\}$$

최악의 경우, 주기 T_S 동안 클라이언트가 수신하게 되는 최대 프레임 수는 다음과 같다.

$$C_R = \frac{S_S}{P_T} = \frac{S_{GR}\{(1.25R+1)Z - 1.5R\sqrt{Z} + R\}}{Z}$$

4. 성능 분석

4.1. 전송 회수 비교

다음 그림 9는 주변 그룹에 대한 전송 비율 $R = 1$인 경우, 영역의 개수에 따른 그룹 단위 전송 방법과 가상 그룹화 방법에서의 서버의 최대 전송 회수를 비교한 것이다.

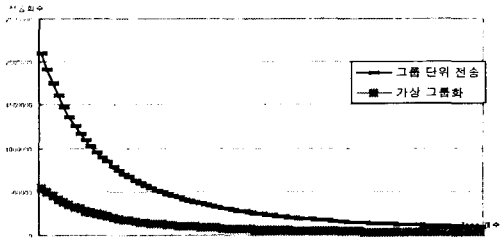

그림 9 영역 개수에 따른 그룹 단위 전송 방법과 가상 그룹화 방법의 서버 전송횟수 비교

위 그래프의 결과를 살펴보면, 영역의 개수가 동일할 때, 가상 그룹화 방법이 그룹 단위 전송 방법보다 전송회수가 평균 1/3.84 정도로 줄어듦을 알 수 있다. 이것은, 가상 그룹화 방법이 적은 수의 프로세스로 그룹 단위 전송 방법과 동일한 효과를 낼 수 있음을 알 수 있다.

프로세스 개수가 늘어나면, 서버의 수행 시간 증가와 서버의 리소스 소모량이 증가하게 된다. 각 프로세스의 리소스 사용량을 K 라고 할 때, 영역의 개수에 따른 총 리소스 사용량 K_T 는 다음과 같다.

$$K_T = K * Z$$

그러므로 영역의 크기가 같을 때, 그룹 단위 전송 방식에서의 리소스 사용량을 K_T^S 라고 하면 가상 그룹화 방식에서의 리소스 사용량은 다음과 같게 되어 훨씬 리소스 소모가 적음을 알 수 있다.

$$K_T^V = \frac{K_T^S}{3.84}$$

또한 위 그래프의 결과를 보면, 영역의 개수가 증가함에 따라 전송 횟수가 줄어들고 있음을 알 수 있다. 하지만, 전송 개수를 줄이기 위해 영역의 개수를 늘리는 경우에는 여러 문제점이 있다. 영역을 최대한 작게 나누면 영역을 관리하기 위한 프로세스의 개수 또한 늘어나게 되고, 또한 영역이 작아지면 바로 옆 영역에 있는 정보만 받아서는 필요한 정보를 모두 수신하지 못하는 문제점이 있다.

다음 그림 10은 영역의 크기가 너무 작아지면 실제 필요한 정보를 모두 수신하지 못함을 보여준다.

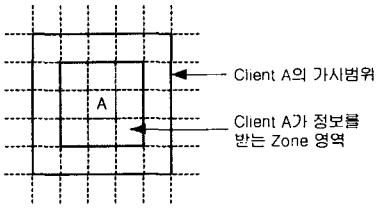

그림 10 영역의 크기가 너무 작은 경우

영역의 크기는 개발하는 응용에 따라 결정되는 것이므로 최적의 영역의 크기는 본 논문에서 다룰 수 있는 범위를 넘어서므로 최적의 영역의 크기는 고려하지 않도록 하고, 영역을 관리하기 위한 프로세스 개수가 늘어날 때 성능에 어떤 영향을 미치는지 다음절에서 비교해보기로 하겠다.

4.2. 영역 개수에 따른 서버 수행 시간 비교

프로세스 개수가 많아지면, 서버가 주기 T_S 마다 수신된 모든 메시지를 처리하는데 걸리는 전체 수행 시간이 길어지게 된다. 클라이언트와 네트워크 상에서 발생하는 지연은 서버 측에서 조절할 수 없는 값이므로, 본 논문에서는 서버 측에서의 지연, 즉 서버의 수행시간만을 분석하도록 하겠다. 서버의 수행 시간을

계산하기 위해서 사용하는 기호는 다음과 같다.

T_{MS} : 프로세스가 한 메시지를 전달하기 위해 처리하는 시간

T_{MP} : 프로세스가 수신한 한 메시지를 처리하는 시간

T_P : 그룹을 관리하는 프로세스의 처리 중 메시지 송수신 이외의 처리시간을 합한 시간.

서버 수행시간 T_{total} 은 다음과 같다.

$$T_{total} = T_P \cdot Z + T_{MP} \cdot P_T + T_{MS} \cdot S_S$$

주변 그룹에 대한 전송 비율을 $R = 1$ 이라 할 때, 그룹 단위 전송 사용 시 전체 수행 시간은 다음과 같다.

$$T_{total} = T_P \cdot Z + T_{MP} \cdot P_T + T_{MS} \cdot \{S_{GR} \cdot P_Z\{9Z - 12\sqrt{Z} + 4\}\}$$

가상 그룹화 방법을 사용할 때의 전체 수행시간은 다음과 같다.

$$T_{total} = T_P \cdot Z + T_{MP} \cdot P_T + T_{MS} \cdot \{S_{GR} \cdot P_Z\{2.25Z - 1.5\sqrt{Z} + 1\}\}$$

그룹 단위 전송 방법과 가상 그룹화 방법의 전체 수행시간을 비교해보면 항상 가상 그룹화 방법의 수행시간이 짧음을 알 수 있다. 그러므로, 영역 개수에 따른 성능 분석은 가상 그룹화 방법을 대상으로 하고, T_{MP}, T_{MS}, T_P 값은 운영 환경과 개발하는 응용에 따라 달라지므로 다음과 같은 경우로 나뉘어 수행하였다. ($P_T = 10000$, $Z = 1 \sim 10,000$)

Case 1 : $T_P = T_{MS} = T_{MP} = 1$

Case 2 : $T_{MS} = T_{MP} = 1$, $T_P = 10 \cdot T_{MS}$

Case 3 : $T_{MS} = T_{MP} = 1$, $T_P = 100 \cdot T_{MS}$

이것을 그래프로 표시하면 다음 그림 11과 같다.

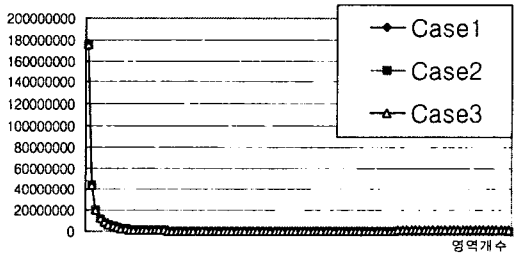

그림 11. 영역의 개수에 따른 성능 분석 (영역 1 ~ 10,000)

그림 11을 보면, 영역의 개수가 작으면, 전송할 데이터 개수가 많기 때문에 시간이 많이 걸림을 알 수 있다. 다음 그림 12는 영역의 개수가 400 ~ 10000 일 때의 수행 시간을 표시한 것이다

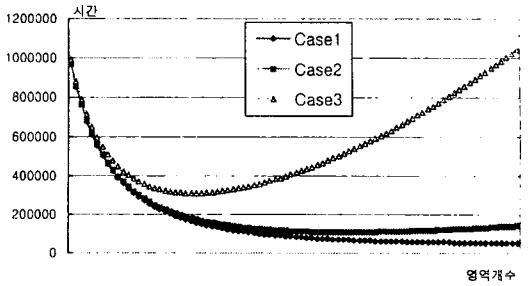

그림 12 영역 개수에 따른 성능 분석 (영역 400 - 10000)

위의 결과를 보면 $T_P = T_{MS} = T_{MP} = 1$ 경우 영역을 작게 나눌수록 성능이 향상됨을 알 수 있고, $T_{MS} = T_{MP} = 1$, $T_P = 100 \cdot T_{MS}$ 경우 영역의 개수가 증가할수록 성능이 나빠짐을 알 수 있다.

5. 결론

본 논문에서는, 네트워크 게임의 지연을 최소화하기 위한 방법으로, 게임상의 맵을 일정 영역으로 분할하여 그룹화한 후, 각 그룹을 4개의 가상 그룹으로 분할하여 전송하게 함으로써, 그룹 단위 전송 방법 보다 전송회수를 1/3.8 정도로 줄여 서버의 부하 및 네트워크 트래픽 감소로 인한 응답 지연 및 응답 손실을 최소화하는 방법을 제안하였다.

앞으로의 연구과제는 특정 영역에 많은 클라이언트가 몰리거나, 특정 영역에 적은 수의 클라이언트가 있는 경우, 클라이언트가 많은 영역은 좀더 작은 영역으로 세분화하고, 클라이언트가 적은 영역들을 하나의 그룹으로 묶어 처리의 효율성을 높이는 방법을 연구해야 할 것이다.

[참고문헌]

[1] Laurent Gautier, Christophe Diot, "Design and Evaluation of MiMaze, a Multi-Player Game on the Internet" IEEE Multimedia Systems Conference, 1998, 6

[2] M.S. Borella, "Source models of network game traffic", IEEE Computer Communications vol 23, 403-410, 2000

[3] Yu-Shen Ng, "Designing Fast-Action Game For The Internet", Gamasutra, 1997,9

[4] 신동원, "OpenMUD", 아주대학교, 1998

[5] Jesse Aronson, "Using Grouping for networked gaming", Game Developer, 1997,8