http://dx.doi.org/10.7236/JIIBC.2013.13.4.29
JIIBC 2013-4-5

멀티플레이어 게임 서버 시스템의 규모조정을 위한 통신 대역폭 요건 감소 기법

Reduction Method of Network Bandwidth Requirement for the Scalability of Multiplayer Game Server Systems

김진환*

Jinhwan Kim

요 약 전형적으로 멀티플레이어 게임은 클라이언트-서버 구조 혹은 peer-to-peer 구조를 기반으로 구성된다. 클라이언트-서버 구조는 플레이어의 수가 증가할 때 서버에서 대역폭 요건이 커지기 때문에 규모조정이 어렵다. 반면 peer-to-peer 구조는 각 플레이어마다 자신의 상태와 다른 모든 플레이어들의 상태 간의 일관성을 점검해야 하기 때문에 상당한 오버헤드가 수반된다. 본 논문에서는 두 구조의 장점을 결합하는 기법을 제시한다. 이 기법에서는 플레이어들이 우선순위가 낮은 갱신 작업은 peer-to-peer 방식으로 상호 교환하며 우선순위가 높은 갱신 작업에 대해서는 중앙 서버와 직접 통신하게 된다. 결과적으로 제시된 기법은 클라이언트-서버 구조 보다 통신 부하 요건이 감소되며 서버의 대역폭 병목 현상도 제거된다. 멀티플레이어 게임의 다른 중요한 사항인 플레이어 간의 상태 일관성도 이 기법에서 항상 정확히 유지된다. 제시된 기법의 성능은 분석과 다양한 실험을 통하여 평가되었다.

Abstract Multiplayer games typically organized based on a client-server(CS) or peer-to-peer(PP) architecture. The CS architecture is not scalable with the number of players due to a large bandwidth requirement at the server. The PP architecture, on the other hand, introduces significant overhead for the players, as each player needs to check the consistency between its local state and the state of all other players. We then propose a method that combines the merits of CS and PP. In this method, players exchange updates with lower priority in a peer-to-peer manner but communicate directly with a central server for the other updates. As a result, the proposed method has a lower network bandwidth requirement than the server of a CS architecture and the server bandwidth bottleneck is removed. For another important issue about multiplayer games, this method always maintains state consistency among players correctly. The performance of this method is evaluated through extensive simulation experiments and ana;ysis.

Key Words : Client-server, Peer-to-peer, Bandwidth, Multiplayer, Consistency, Scalability

Ⅰ. 서 론

인기가 최근 급격히 증가하고 있는 멀티플레이어 게임은 대부분 사람들과의 상호 작용이 포함되기 때문에 대역폭 관리[1]와 상태 일관성(consistency)에 대한 복잡도 문제가 수반된다[2]. 통신 대역폭 요건은 게임의 규모

*정회원, 한성대학교 멀티미디어공학과
접수일자 : 2013년 3월 8일, 수정완료 : 2013년 7월 5일
게재확정일자 : 2013년 8월 16일

Received: 8 March, 2013 / Revised: 5 July, 2013
Accepted: 16 August, 2013
*Corresponding Author: kimjh@hansung.ac.kr
Dept. of Multimedia Engineering, Hansung University, Korea

조정과 직결된다. 예를 들면 게임 서버의 대역폭 요건은 플레이어 수의 제곱에 비례하여 증가하기 때문에 참여하는 플레이어의 수가 증가하면 서버에 대한 통신 대역폭 병목 현상이 쉽게 발생할 수 있다. 멀티플레이어 게임에 관한 다른 중요한 사항인 플레이어 간의 상태 일관성도 정확히 유지될 필요가 있다. 예를 들어 자동차 경주 게임에서 모든 플레이어는 각 자동차의 위치에 관한 동일한 화면을 볼 수 있어야 하며 1 인칭 shooting 게임에서는 모든 플레이어들이 누가 죽었고 누가 살아 있는 지를 동일하게 파악할 수 있어야 한다.

멀티플레이어 게임은 전형적으로 클라이언트-서버(CS; Client-Server) 또는 Peer-to-Peer(PP) 구조를 기반으로 구성된다. 본 논문에서는 클라이언트-서버 구조를 이하 CS 구조로 Peer-to-Peer 구조를 PP 구조로 기술한다. CS 구조에서는 플레이어들이 상태 일관성을 항상 책임지는 중앙 서버를 통하여 주기적인 갱신 작업을 상호 교환한다. 반면 PP 구조에서는 상태 일관성이 해결되는 복잡한 분산 협정 프로토콜이[3] 사전에 설정되어 있어야 하며 이를 근거로 각 플레이어가 다른 모든 플레이어와 직접 통신하게 된다. CS 구조는 플레이어의 수가 증가할 때 서버의 대역폭 요건이 플레이어의 수 제곱에 비례하여 증가되기 때문에 규모 조정이 어렵다. 대신 CS 구조는 상태 일관성을 단순하게 유지하며 보장할 수 있는 장점이 있다. PP 구조에서는 서버의 대역폭 요건에 대한 병목 현상은 없는 대신 상태 일관성을 해결하기 위한 복잡한 분산 협정 프로토콜이 반드시 설정되어야 하는 단점이 있다. 즉 각 플레이어는 자신의 지역적 상태와 다른 모든 플레이어들의 상태 간의 일관성을 항상 점검해야 하기 때문에 상당한 오버헤드가 수반된다.

본 논문에서는 멀티플레이어 게임을 위한 두 구조의 장점을 모두 사용하여 상태 일관성을 항상 정확히 유지하면서 CS 구조의 서버에서 발생하는 대역폭 요건을 감소시키는 기법을 제시한다. 멀티플레이어 게임 시스템에서 클라이언트가 발생하는 이벤트는 게임 시스템의 실시간적 요건과 중요도에 따라 우선순위가 설정될 수 있다[4]. 본 논문에서는 우선순위가 낮은 이벤트는 플레이어들이 PP 방식으로 상호 교환하며 우선순위가 높은 이벤트는 플레이어와 서버가 CS 방식처럼 통신하는 기법을 제시한다.

본 논문은 2 장에서 CS 구조와 PP 구조의 특성과 대역폭 요건이 기술되며 3장에서 우선순위별로 그룹화된 이벤트의 특성과 두 구조의 장점을 사용하는 통신 대역폭 감소 기법이 기술된다. 그리고 4 장에서 분석과 시뮬레이션을 통하여 다른 기법과 비교된 성능을 분석하며 5 장에서는 결론을 기술한다.

II. CS 구조와 PP 구조의 대역폭 요건

1. CS 구조

가. CS 구조 특성

멀티플레이어 인터넷 게임을 위한 가장 일반적인 구조는 CS 모델이다(그림 1 참조). 이 구조에서 특정 호스트 혹은 특정 플레이어는 다른 플레이어들이 연결된 서버를 통하여 설정된다[5]. 상태 처리와 같은 게임 시뮬레이션은 클라이언트보다는 주로 서버에서 수행되며 각 플레이어는 새로운 액션을 이벤트 형식으로 서버에 전송한다. 서버는 새로운 시뮬레이션을 수행하기 위하여 수신한 정보를 사용하며 시뮬레이션 결과 상태를 플레이어에게 다시 전송하게 되고 각 플레이어는 갱신된 게임 화면을 보게 된다.

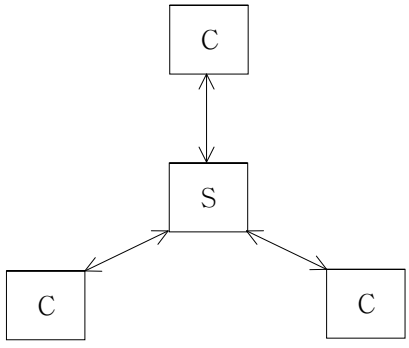

그림 1. CS 구조
Fig. 1. CS architecture

CS 구조는 구현이 비교적 단순하며 집중화된 서버를 통하여 상태 일관성을 유지하기가 수월하고 플레이어의 속임수 행위가 어렵다. 또한 게임 제공업체는 영리를 추구할 수 있는 플레이어 가입 구조를 생성하고 제어할 수 있으며 지속적인 게임 감시도 가능하다. 이러한 이유 때문에 CS 구조가 멀티플레이어 게임에 많이 활용되고 있다.

그러나 CS 구조는 여러 문제점들이 존재한다. 첫째, 서버의 결함이 발생할 경우 게임 수행 자체가 불가능하게 된다[6, 7]. 둘째, 플레이어 측면에서의 대역폭은 크지

않지만 서버 측면에서의 대역폭은 매우 커질 수 있다[7, 8]. 결과적으로 서버는 대용량의 통신 대역폭이 필요하며 서버를 통해 플레이어들에게 전달되는 메시지들은 통신 지연시간이 다소 증가될 수 있다. 이러한 통신 지연 시간은 상태의 비일관성을 해결하는 시간을 증가시킬 수 있다.

나. 대역폭 요건

특정 클라이언트는 플레이어의 갱신 내용을 플레이의 갱신 주기마다 서버에게 전송하며 이러한 전송은 클라이언트의 상향(upstream) 대역폭 요건으로 정의된다. 본 논문에서 1 초 동안 클라이언트가 서버에게 전송하는 평균 메시지의 수를 n이라 하고 메시지의 평균 크기(단위 byte)를 m이라 할 때 1 초 동안 전송되는 평균 상향 대역폭의 크기는 $n \cdot m$이다.

각 클라이언트의 하향(downstream) 대역폭 요건은 1 초 동안 서버가 N 플레이어들에게 새로운 상태를 표현하는 전역 상태 메시지를 전송하는 크기에 의해 결정된다. 전역 상태 메시지의 크기 L_G는 평균 상향 대역폭과 플레이어의 수의 곱 즉 $N \cdot n \cdot m$ 바이트이며 L_G는 각 클라이언트의 하향 대역폭 요건이 된다. CS 구조에서 각 플레이어에 대한 상향과 하향 대역폭이 모두 포함된 요건은 $n \cdot m+N \cdot n \cdot m$이며 이는 $(N+1) \cdot n \cdot m$이므로 플레이어의 수 N에 비례하게 된다.

$$n \cdot m+N \cdot n \cdot m =(N+1) \cdot n \cdot m \qquad (1)$$

서버는 각 플레이어의 갱신 상태를 수신한 후 게임 상태를 변경하며 모든 플레이어들에게 전역 상태 메시지를 전송하게 된다. 이 과정에서 각 플레이어의 갱신 상태를 수신한 시각과 모든 플레이어들에게 전역 상태 메시지를 전송하는 시각 간의 차이를 서버 지연시간으로 정의한다. 서버는 매초마다 모든 플레이어로부터 갱신 내용을 수신하기 때문에 하향 대역폭 요건은 $N \cdot n \cdot m$이다. 그리고 서버는 매초마다 모든 플레이어에게 전역 상태 메시지를 전송하기 때문에 상향 대역폭 요건은 NL_G이다. 서버의 하향과 상향 대역폭이 포함된 요건은 다음과 같다.

$$N \cdot n \cdot m+NL_G=N(N+1) \cdot n \cdot m \qquad (2)$$

따라서 서버의 대역폭 요건은 플레이들의 수 N^2에 비례하여 증가하게 된다. 서버의 통신량을 감소시키기 위한 interest filtering 기법[9]들이 사용될 수 있으나 이에 대한 구체적 사항들은 본 논문에서 생략하기로 한다.

다. 비일관성 해결

시간 t에서 임의의 플레이어 i의 게임 상태 $S_i(t)$는 플레이어 i가 시간 t에서 보는 게임 화면을 완전하게 기술할 수 있는 속성들의 집합을 의미한다. 전형적으로 게임의 상태는 참여하고 있는 플레이어들 각각의 위치, 방향, 속도. 가속도, 관련 객체 등을 포함한다[5]. 시간 t에서 임의의 두 플레이어들 (i,j)에 대하여 $S_i(t)=S_j(t)$일 경우 이상적으로 모든 플레이어들은 동일한 게임 상태를 유지하는 것으로 볼 수 있다. 그러나 플레이어들 간에 통신망을 통한 지연 시간이 존재하며 플레이어 i와 플레이어 j 간의 통신 지연 시간을 $D_{i,j}$로 정의할 때 시간 t_0에서 플레이어 i의 행위는 플레이어 j에게 시간 $t_0+D_{i,j}$에 알려지게 된다. 이러한 통신 지연 시간 때문에 모든 플레이어들에게 상태 일관성이 항상 정확히 유지되도록 하는 것은 매우 어려운 문제이다. 따라서 두 플레이어들 간에 비일관적 상태가 존재하는 시간 간격을 최소화할 필요가 있다.

실제로 CS 구조에서 서버는 플레이어들 간에 발생하는 상태 비일관성을 모두 파악하고 해결할 수 있다. 예를 들어 플레이어 i가 시간 t_0에서 X 위치로 이동할 경우 이 위치를 이미 플레이어 j가 차지하고 있다면 상태 비일관성이 발생하게 되는 것이다. 서버가 플레이어 i의 최근 이동 요청 사항이 거절된 것을 플레이어 i에게 알려줄 때까지 상태 비일관성이 지속될 수 있다. 따라서 이 경우 비일관성 해결 시간은 서버와 플레이어 간의 통신망에 대한 왕복 지연 시간과 연관된다[10].

본 논문에서는 패킷 손실이 없고 통신 지연 시간의 최대치를 가정한 통신망에서 임의의 플레이어가 비일관적 상태를 가질 수 있는 최대 시간 T_I를 다음과 같이 가정한다.

$$T_I=\max_{i \in I}\{T_S+R_{i,S}\} \qquad (3)$$

CS 구조에서 T_I는 플레이어 i와 서버 S 간의 왕복 지연 시간 $R_{i,S}$와 서버 지연 시간 T_S의 합으로 구성된다(수식 (3) 참조). 이는 서버가 임의의 플레이어 i로부터 메시지를 수신한 시간부터 메시지를 처리하고 해당 플레이어에게 메시지 처리 결과를 다시 전송할 때까지의 시간 중 가장 큰 시간을 의미한다. 모든 플레이어들 간에 발생한 비일관성을 서버가 이 T_I 시간 내에 해결할 수 있다고 가정한다. 실제 멀티플레이어 게임에서 상태 일관성 유지를 위한 bucket 동기화(synchronization) 기법이[11, 12] 제시되었고 이 기법은 수십 millisecond의 정확성을 갖는

클럭 동기화에 기반하고 있다. 본 논문에서 이 기법에 관한 구체적 설명은 기술하지 않기로 한다.

2. PP 구조

가. PP 구조 특성

PP 게임 모델에서는 각 플레이어(peer를 의미함)가 자신의 게임 시뮬레이션을 직접 실행할 책임을 가지고 있다[5]. 그러나 플레이어들 간에 비일관성을 탐지하고 이를 해결하는 서버가 없기 때문에 임의의 비일관성은 분산 협정 프로토콜을 사용하는 플레이어들에 의해 직접 탐지되어야만 한다(그림 2 참조). 예를 들면 분산 게임에서 비일관성을 해결하기 위하여 trailing state[13] 또는 bucket 동기화[12] 기법들이 제시된 바 있다. 이 방법들에 대한 중요한 사항은 정확한 클럭 동기화에 의존하고 있다는 것이다. 플레이어 호스트가 NTP(Network Time Protocol)을 수행하더라 수십 millisecond의 정확성이 유지되는 상업적 인터넷에 연결된 컴퓨터의 클럭들을 NTP가 정확히 동기화할 수 있는 지는 불분명하다[5].

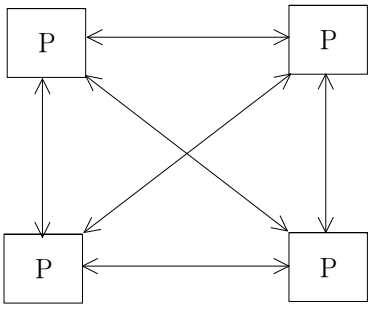

그림 2. PP 구조
Fig. 2. Peer-to-peer architecture

집중화된 서버의 부재는 PP 구조에서 클라이언트 간 메시지 지연 시간의 감소[6, 12], 서버의 결함으로 인한 시스템 중지 사유 소멸 등 여러 가지 이점을 제공한다. 가장 중요한 것은 PP 구조는 서버의 병목 현상이 없다는 것이다. PP 구조가 CS 구조보다 플레이어 측면에서 더 높은 대역폭이 요구되긴 하나 이 요건은 플레이어의 수에 비례하여 증가된다. 그러나 PP 구조는 게임 산업에서 CS 구조처럼 보편적으로 사용되는 상황은 아직 아니다. 중요한 이유는 PP 구조가 CS 구조보다 구현이나 구성이 더 어렵기 때문이다. 또한 PP 구조는 게임에 대한 제어를 사용자들에게 많이 양도하고 있다. 서버의 부재는 게임을 수행하는 사람과 게임 실행 시간 등에 대한 제어가 불가능함을 의미하며 가입자를 통해 이윤을 추구하려는 게임 업체를 어렵게 만드는 상황이 발생한다.

나. 대역폭 요건

PP 구조에서 각 플레이어는 다른 모든 플레이어들과 주기적으로 갱신 내용을 상호 교환하게 되므로 1 초 동안 상향 대역폭 요건과 하향 대역폭 요건은 (N-1)・n・m로 동일하다. 즉 각 플레이어의 대역폭 요건은 다음과 같이 구성된다.

$$2(N-1)\cdot n\cdot m \approx 2N\cdot n\cdot m \qquad (4)$$

결과적으로 PP 구조에서 각 플레이어의 대역폭 요건은 CS 구조의 약 두 배가 된다(수식 (1) 참조). PP 구조는 서버의 대역폭 병목 현상을 제거할 수 있는 반면 플레이어의 대역폭 요건은 플레이어의 수 N에 비례하여 증가하게 된다. 이 구조에서도 interest filtering 기법을 적용하면 대역폭 요건을 상당부분 감소시킬 수 있다.

다. 비일관성 해결

PP 구조에서 플레이어 j가 자신의 상태와 일치하지 않는 갱신 메시지를 플레이어 i로부터 수신할 때 비일관성을 탐지할 수 있다. 이 경우 플레이어 j는 플레이어 i를 포함한 모든 플레이어들에게 갱신-거절 메시지를 전송하게 된다. 이 메시지의 목적은 플레이어 i의 최신 갱신을 무효화하기 위한 것이다. 플레이어 i는 이 메시지를 수신하는 즉시 이전 상태로 복귀(rollback)해야 한다. 이 경우 패킷 손실이 없고 통신 지연 시간의 최대치가 존재하는 상황을 가정할 때 임의의 플레이어 i와 j 간의 왕복 지연 시간이 비일관적 상태를 가질 수 있는 최대 시간 T_I로 결정된다.

$$T_I = \max_{i,j}\{R_{i,j}\} \qquad (5)$$

즉 PP 구조에서 비일관성 해결 시간은 Ri,j와 Ri,S 시간 차이에 따라 CS 구조보다 다소 작은 것으로 분석되었다[5].

III. 서버의 대역폭 요건 감소 기법

1. 이벤트의 우선순위 설정

게임 세계에서 발생하는 이벤트의 본질에 따라 이벤

트들은 상이한 일관성 요건을 가지게 된다[4]. 실시간 이벤트는 엄격한 시간 요건을 갖는 반면 다소 느슨한 일관성 요건을 갖게 되며 반대로 일관성 이벤트는 느슨한 시간 요건과 엄격한 일관성 요건을 갖게 된다. 엄격한 시간 요건과 일관성 요건을 모두 가지는 일관성 실시간 이벤트는 가장 중요한 이벤트로 간주된다. 사람이 이벤트에 대한 결과를 감지할 수 있는 시간인 100ms보다 큰 이벤트들은 일관성 이벤트로 분류되며 이보다 작은 시간을 갖으면 실시간 이벤트 또는 일관성 실시간 이벤트로 분류된다[4]. 실제 대규모 멀티플레이어 게임 Quake[14] 명령어인 이벤트들 중 Fire, Impact 이벤트들은 일관성 실시간 요건을 가지며, Move 이벤트는 실시간 요건만 가지고 Damage/die, Spawn 이벤트들은 일관성 요건만 가지는 것으로 분류된다.

상이한 이벤트들의 요건들은 게임 서버의 CPU 처리를 위한 스케쥴링과 전송 스케쥴링 과정에서 상이한 우선순위로 설정될 수 있다. 본 논문에서는 일관성 실시간 이벤트인 Quake의 Fire, Impact 이벤트들의 우선순위를 가장 높은 1 그룹으로 설정하며 실시간 이벤트인 Move 이벤트를 다음 우선순위인 2 그룹으로 설정한다. 그리고 실시간 요건이 없는 일관성 이벤트인 Damage/die와 Spawn 이벤트를 가장 낮은 우선순위인 3 그룹으로 설정한다.

2. 우선순위 기반 대역폭 요건 감소 기법

가. 하이브리드 CS-PP 구조

멀티플레이어 게임에서 각 플레이어는 CS 구조에서 클라이언트 역할을 수행하며 서버에게 이벤트를 전송하게 된다. 본 논문에서는 클라이언트가 1 초 동안 서버에게 전송하는 평균 이벤트 또는 메시지의 수를 n이라 하며 1, 2, 3 그룹의 평균 이벤트 수를 각각 n_1, n_2, n_3로 정의한다. 즉 수식 (6)과 같이 n은 n_1, n_2, n_3의 합이 된다.

$$n=n_1+n_2+n_3 \quad (6)$$

CS 구조에서 서버의 대역폭 병목 현상을 감소시키고자 우선순위가 높은 1 그룹과 2 그룹의 이벤트들은 클라이언트가 서버에게 전송하고 우선순위가 낮은 3 그룹의 이벤트들은 PP 구조처럼 클라이언트들이 서버를 통하지 않고 직접 상호 교환하는 방식을 제시한다(그림 3 참조).

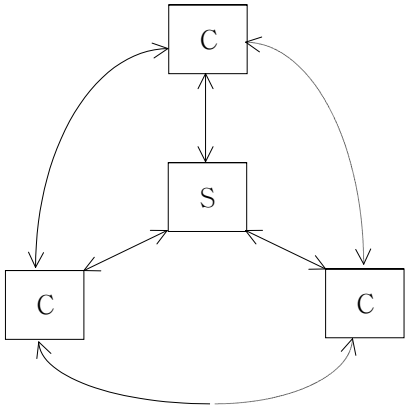

그림 3. 하이브리드 CS-PP 구조
Fig. 3. Hybrid CS-PP architecture

나. 대역폭 요건

그림 3의 하이브리드 CS-PP 구조에서 클라이언트가 1 그룹과 2 그룹의 이벤트들을 서버에게 전송할 경우 매초 $(n_1+n_2)\cdot m$ 대역폭이 필요하며 3 그룹의 이벤트를 다른 모든 클라이언트들에게 전송할 경우 매초 $n_3\cdot(N-1)m$ 대역폭이 필요하다. 따라서 클라이언트의 상향 대역폭 요건은 수식 (7)과 같다.

$$(n_1+n_2)\cdot m+n_3\cdot(N-1)\cdot m \quad (7)$$

이 구조에서 클라이언트는 매초 서버로부터 $N\cdot(n_1+n_2)\cdot m$ 대역폭에 해당하는 전역 상태 메시지를 수신하게 되며 다른 모든 클라이언트들로부터 $n_3\cdot(N-1)\cdot m$ 대역폭의 메시지를 수신하게 되므로 하향 대역폭 요건은 수식 (8)과 같이 설정된다.

$$N\cdot(n_1+n_2)\cdot m+n_3\cdot(N-1)\cdot m \quad (8)$$

따라서 클라이언트의 하향과 상향 대역폭을 포함한 요건은 다음과 같다.

$$(n_1+n_2)\cdot(N+1)\cdot m+2n_3\cdot(N-1)\cdot m \quad (9)$$

이 구조에서 서버는 매초 1 그룹과 2 그룹의 이벤트들을 N 플레이어로부터 수신하기 때문에 하향 대역폭 요건은 $N\cdot(n_1+n_2)\cdot m$이며 게임 세계의 전역 상태 메시지를 N 플레이어들에게 다시 전송하는 상향 대역폭 요건은 $N^2\cdot(n_1+n_2)\cdot m$이 된다. 따라서 서버의 상향과 하향 대역폭이 모두 포함된 요건은 다음과 같다.

$$N \cdot (n_1+n_2) \cdot m+N^2 \cdot (n_1+n_2) \cdot m$$
$$=N \cdot (N+1) \cdot (n_1+n_2) \cdot m \quad (10)$$

이 경우 하이브리드 CS-PP 구조의 서버 대역폭은 CS 구조의 서버 대역폭(수식 (2) 참조)에 비하여 n_3/n만큼 감소하게 된다. 하이브리드 CS-PP 구조에서 서버가 1 그룹의 이벤트들만 클라이언트로부터 수신하고 2 그룹과 3 그룹의 이벤트들을 클라이언트들이 직접 상호 교환할 경우 서버의 대역폭 요건은 수식 (11)과 같이 설정된다.

$$N \cdot n_1 \cdot m+N^2 \cdot n_1 \cdot m$$
$$=N \cdot (N+1) \cdot n_1 \cdot m \quad (11)$$

이 경우 하이브리드 CS-PP 구조의 서버 대역폭은 CS 구조의 서버 대역폭에 비하여 $(n_2+n_3)/n$만큼 감소하게 된다. 제시된 CS-PP 구조에서도 서버의 대역폭은 플레이어들의 수 N^2에 비례하여 증가하나 CS 구조의 서버 대역폭 보다는 n_3/n 또는 $(n_2+n_3)/n$만큼 감소되기 때문에 서버의 병목 현상을 완화할 수 있다. 단 1 그룹과 2 그룹의 이벤트들을 서버에게 전송할 경우 서버의 대역폭이 감소되는 만큼 클라이언트들의 대역폭은 수식 (9)와 수식 (1)의 차이만큼 증가하게 된다. 결과적으로 CS 구조, PP 구조, 하이브리드 CS-PP 구조 모두 필요한 전체 통신 대역폭은 거의 동일하기 때문에 서버의 대역폭이 감소되는 대신 클라이언트의 대역폭이 증가하게 된다. 표 1은 세 가지 구조의 클라이언트 C와 서버 S의 대역폭을 수식으로 비교한 결과이며 하이브리드 CS-PP 구조는 1 그룹과 2 그룹의 이벤트들만 서버가 수신하는 상황을 가정한 것이다.

표 1. 구조별 통신 대역폭
Table 1. Network Bandwidth of Architectures

대역폭 / 구조	C	S
CS	(N+1)nm	N(N+1)nm
하이브리드 CS-PP	$(n_1+n_2)(N+1)m$ $+2n_3(N-1)m$	$N(N+1)(n_1+n_2)m$
PP	2(N-1)nm	0

3. 비일관성 해결

하이브리드 CS-PP 구조에서는 우선순위가 높은 그룹의 메시지를 서버가 클라이언트로부터 수신하며 우선순위가 낮은 그룹의 메시지들은 클라이언트들이 직접 상호 교환하는 방식이 적용되기 때문에 두 구조의 비일관성 해결 방법이 모두 적용된다. 즉 1 그룹과 2 그룹의 메시지들이 클라이언트로부터 서버로 전송될 경우 서버는 전형적인 CS 구조의 비일관성 해결 방법을 사용하며 3 그룹의 메시지들에 대해서는 클라이언트가 직접 비일관성을 해결하는 PP 구조의 방법을 사용한다.

IV. 성능 분석

1. 실험 환경과 변수

본 논문에서는 대규모 멀티플레이어 Quake 게임의 각 플레이어가 1초 마다 평균 n(=3) 개의 이벤트를 발생하며 각 우선순위 그룹의 이벤트가 한 개씩 동일한 비율로 구성된 것을 가정한다.

$$n=n_1+n_2+n_3=3$$
$$n_1=n_2=n_3=1$$

플레이어가 발생한 이벤트는 메시지로 서버에게 전송되며 평균 크기 m은 20 바이트이고 최소값 10 바이트와 최대값 30 바이트 분포 범위에서 실제 메시지의 크기가 설정된다. 플레이어의 수는 300명부터 800명까지 100명씩 증가시킨 상태에서 10 분간 측정된 서버의 평균 대역폭을 Kbyte(=1024 바이트) 단위로 측정하였다. 본 논문의 하이브리드 CS-PP 구조에서 1 그룹과 2 그룹의 메시지들만 서버가 처리한 대역폭은 CS-PP(1, 2 그룹)로 표기되며 1 그룹의 메시지들만 서버가 처리한 결과는 CS-PP(1 그룹)으로 표기되어 그룹별 구분이 없는 CS 구조의 서버 대역폭과 그림 4에서 비교 분석되었다.

PP 구조는 서버가 존재하지 않기 때문에 그림 4에서 PP 구조의 결과는 기술되지 않는다. CS 구조의 서버는 우선순위 그룹에 상관없이 모든 플레이어의 이벤트를 처리해야 하는 반면 하이브리드 CS-PP 구조의 서버는 우선순위가 높은 그룹의 이벤트만을 처리하기 때문에 서버의 대역폭이 감소된다. 1 그룹과 2 그룹의 이벤트들을 처리하는 서버의 대역폭은 CS 구조의 서버보다 약 33.33% 감소되며 1 그룹만의 이벤트만을 처리하는 서버의 대역폭은 CS 구조의 서버보다 약 66.67% 정도 감소된 결과가 그림 4에서 나타났다.

따라서 하이브리드 CS-PP 구조에서는 감소된 서버의 대역폭만큼 플레이어의 수를 증가되거나 플레이어의 초

당 평균 이벤트 수가 증가되는 상황을 수용할 수 있다. 예를 들어 CS 구조의 서버가 800명을 수용할 수 있는 대역폭을 하이브리드 CS-PP 구조의 서버가 사용할 경우 1 그룹과 2 그룹의 이벤트를 처리할 때는 약 266명이 증가된 1066명의 플레이어들을 수용할 수 있고 1 그룹의 이벤트만을 처리할 때는 약 533명이 증가된 1333명의 플레이어들을 수용할 수 있다.

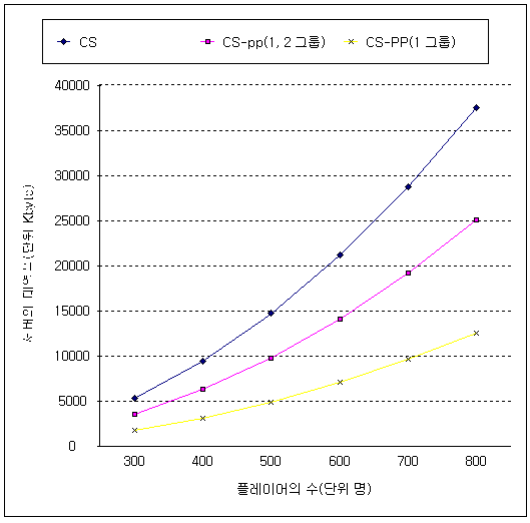

그림 4. 서버의 대역폭
Fig. 4. Bandwidth of the server

그림 5에서는 PP 구조를 포함하여 각 플레이어의 평균 대역폭(단위 Kbyte)이 비교되고 있다. CS 구조에서 플레이어의 대역폭이 가장 작은 것으로 나타났으며 이는 다른 구조에 비하여 서버의 대역폭이 훨씬 더 크기 때문인 것으로 분석된다. 하이브리드 CS-PP 구조에서 서버가 1 그룹과 2 그룹의 이벤트들을 수신하는 경우 플레이어는 그룹 3의 이벤트들만 다른 플레이어들과 상호 교환하기 때문에 플레이어의 대역폭이 CS 구조보다는 크게 나타났다. 그리고 이 대역폭은 서버가 1 그룹의 이벤트들만 수신하고 플레이어들이 2 그룹과 3 그룹의 이벤트들을 직접 상호 교환해야 할 경우의 플레이어 대역폭보다는 작은 것으로 나타났다. 즉 이 경우 서버의 대역폭이 감소되는 반면 플레이어의 대역폭이 증가되는 것으로 분석된다. 서버없이 플레이어들만 메시지를 상호 교환하는 PP 구조에서는 플레이어의 대역폭이 가장 크며 이는 CS 구조보다 거의 두 배 정도인 것으로 측정되었다. 그림 5에 나타난 바와 같이 각 구조에서 공통적으로 플레이어의 대역폭은 플레이어의 수에 비례하여 증가하는 것으로 나타났다.

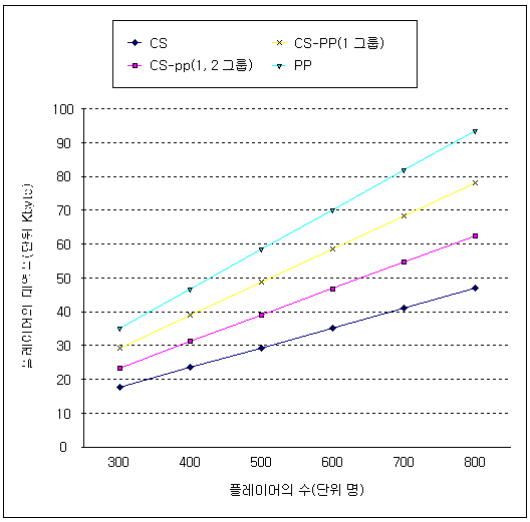

그림 5. 플레이어의 대역폭
Fig. 5. Bandwidth of the player

V. 결론

대규모의 플레이어들이 참여하는 멀티플레이어 게임에서 서버의 통신 대역폭을 감소시키기 위하여 하이브리드 CS-PP 구조를 본 논문에서 제시하였다. CS 구조의 서버는 플레이어 수의 제곱에 비례하는 통신 대역폭 특성상 병목 현상이 쉽게 발생할 수 있다. 서버가 없는 PP 구조에서는 플레이어의 대역폭이 CS 구조의 플레이어보다 두 배 정도로 증가되며 상태 일관성을 유지하기 위한 복잡한 분산 협정 프로토콜이 필요하다. 본 논문에서는 플레이어의 이벤트 특성에 연관된 우선순위를 활용하여 우선순위가 높은 이벤트들을 CS 구조의 서버가 처리하고 우선순위가 낮은 이벤트들은 플레이어들이 직접 상호 교환하는 PP 구조의 기법을 모두 활용하여 서버의 대역폭을 감소시킴으로써 병목 현상이 발생되지 않도록 하였다.

제시된 하이브리드 CS-PP 구조의 서버에서 감소된 대역폭만큼 증가된 플레이어들을 수용할 수 있으며 일시적 과부하를 해결할 수 있다. 또한 CS 구조와 PP 구조의 일관성 기법을 모두 사용함으로써 게임 시스템 전체의 일관성이[15] 정확히 유지될 수 있다. 제시된 구조와 기법은 향후 모바일 환경의[16] 대규모 멀티플레이어 게임시스

템의 요건 충족 및 성능 향상을 위하여 개선될 것으로 기대된다.

References

[1] P. Koutsakis, M. Vafiadis, and A. Lazaris, "A New Bandwidth Allocation Mechanism for Next Generation Wireless Cellular Networks," Wireless Network 16, pp. 331-353, 2010.Nov. 2009.

[2] Y. W. Ahn, A. M. K. Cheng, J. Baek, and P. S. Fisher, "A Multiplayer Real-Time Game Protocol Architecture for Reducing Network Latency," IEEE Transactions on Consumer Electronics 55(4), pp. 1883 - 1889,

[3] G. Coulouris, and J. Dolimore, and T. Kindberg. "Distributed Systems Concepts and Design"". Addison-Wesley, 2001.

[4] A. Hsu, J. Ling, and Q. Li, and C. C. Jay Kuo, "On the design of Multiplayer On-line Video Game Systems," SPIE ITCOM, Sep. 2003.

[5] J. D. Pellegrino and C. Dovrolis, "Bandwidth Requirement and State Consistency in Three Multiplayer Game Architectures," 2nd Workshop on Network and System Support for Games, pp. 52-59, 2003.

[6] P. Bettner and M. Terrano. "1500 Archers on a 28.8 Programming in Ages of Empires and Beyond," Technical report, Ensemble Studios, 2001.

[7] M. Mauve. "How to Keep a Dead Man from Shooting", 7th International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services, pp. 199-204, October 2000.

[8] E. Cronin, B. Filstrup, and A. Kurc. "A Distributed Multi-Player Game Server System," EECS589, Course Project Report, University of Michigan, May 2001.

[9] L. Zou, M. H. Ammar, and C. Diot. "An Evaluation of Grouping Techniques for State Dissemination in Networked Multi-User Games," 9th International Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems, August 2001.

[10] S. Harcsik, A. Petlund, C. Griwods, and P. Halvorsen, "Latency Evaluation of Networking Mechanisms for Game Traffic," 6th Workshop on Networks and System Support for Games, pp. 129-134, Sep. 2007.

[11] L. Gautier and C. Diot and J. Kurose. "End-to-End Transmission Control Mechanisms for Multiparty Interactive Applications on the Internet," 18th Annual Joint Conference of the IEEE Computer and Communications Societies, pp. 1470-1479, Mar. 1999.

[12] C. Diot and L. Gautier. "A Distributed Architecture for MultiPlayer Interactive Applications on the Internet," Network, IEEE 13(4), August 1999.

[13] E. Cronin, B. Filstrup, and A. Kurc and S. Jamin. "An Efficient Synchronization Mechanism for Mirrored Game Architectures," 1st Workshop on Network and System Support for Games, pp. 67-73, April 2002.

[14] Doom, Quake, ID Software, Inc. http://www.idsoftware.com

[15] S. Kang, Y. Kim, J. Lee, "Implementation of Synchronous System between Real-Robot and Virtual-Robot for Smart Device-based Fighting Game," Journal of the Korea Academia-Industrial cooperation Society, v. 10, no. 8, pp. 1-217, Aug. 2012.

[16] J. Kim, J. Jung, "Implementation of Wireless Communication Module with Point-to-multipoint Media Access Control," Journal of the Institute of Webcasting, Internet and Telecommunication, v. 12, no. 5, pp. 267-274, Oct. 2012.

※ 본 연구는 한성대학교 교내연구비 지원과제임.

저자 소개

김 진 환(정회원)

- 1986년 : 서울대학교 컴퓨터공학과 졸업(학사)
- 1988년 : 서울대학교 컴퓨터공학과 졸업(석사)
- 1994년 : 서울대학교 컴퓨터공학과 졸업(박사)
- 1994년~1995년 : 서울대학교 컴퓨터 신기술공동연구소 특별연구원
- 1995년~현재 : 한성대학교 멀티미디어공학과 교수

<주관심분야 : 멀티미디어 시스템, 실시간 게임 시스템>