온라인 퀴즈 게임에서 공정성을 제공하기 위한 게임 시나리오

김건웅*, 윤성중*
*목포해양대학교 해양전자・통신공학부
e-mail: kgu@mmu.ac.kr

A Game Scenario to Support Fairness in On-line Quiz Game

Geonung Kim*, Sung-Jung Yoon*
*Division of Marine Electronic & Communication Engineering,
Mokpo National Maritime University

요 약

온라인 퀴즈 쇼와 같은 게임에서 참여자들의 서로 다른 망 환경과 컴퓨팅 환경으로 인해 공정성을 보장할 수 없는 문제가 발생할 수 있다. 본 논문에서는 환경의 차이로 인한 불공정성을 개선할 수 있는 게임 진행 시나리오를 소개한다. 제안한 시나리오에서는 클라이언트에서 질문이 도착하면, 질문 도착 시간을 저장하고, 사용자가 응답을 하는 경우, 사용자의 답과 질문 도착 후 경과된 시간을 같이 서버에게 전달하고, 서버에서는 응답이 도착한 순서가 아닌, 질문에 대한 응답을 만들어낸 소요 시간을 기준으로 승자를 판정하도록 하여 망 환경과 연산 환경의 차이로 인한 불공정성을 제거한다.

1. 서론

WWW의 등장과 활성화는 과거 연구나 학술 분야 등 특정 분야의 종사자들 사이에서 자원이나 정보의 공유 수단으로 이용되었던 인터넷을 일반인들도 쉽게 접하고 이용할 수 있는 매체로 탈바꿈시켰다. 특히 인터넷을 이용한 온라인 게임의 확산은 눈부시며, 게임은 현재 IT 산업의 킬러 애플리케이션 중 하나로 자리매김했다. 현재 과거의 제한된 환경에서 진행되던 많은 게임 형식이 인터넷에 맞게 변형되어 등장하고 있으며, 그 대표적인 예로 온라인 RPG(Roll Playing Game), 온라인 바둑, 온라인 카지노(Casino) 등을 들 수 있다.

우리는 다수의 사용자에게 질문을 던지고 정답을 빨리 말하는 사람이 점수를 얻는 방식의 퀴즈 게임을 자주 보게 된다. 본 논문에서는 이러한 방식의 퀴즈 게임을 온라인상에서 구현하고자 했을 때 만날 수 있는 불공정성 문제를 살펴보고, 이러한 불공정성을 제거할 수 있는 게임 진행 시나리오를 제안한다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 퀴즈 쇼를 온라인상에서 진행하고자 했을 때 만날 수 있는 불공정성 문제를 살펴보고, 3장에서는 망 환경 차이로 인한 불공정성 문제를 해결할 수 있는 알고리즘들을 제안한다. 4장에서는 제안된 알고리즘을 적용한 클라이언트와 서버의 게임 진행 시나리오를 보이고 5장에서 결론을 맺는다.

2. 온라인 퀴즈 게임에서의 불공정성 문제

오프라인에서의 퀴즈 게임에서는 모든 참여자가 동시에 질문을 듣고 질문의 답을 알고 있는 참여자가 바로 부저를 울리거나 정답을 올려서 빨리 정답을 맞힌 사용자가 점수를 얻는 진행 방식이 대부분이다. 이를 온라인상에서 구현하고자 하면, 문제를 만들어 전송하고 각 사용자들의 응답을 받아 승자를 판정하는 서버와 사용자들에게 질문을 보여주고 사용자가 입력한 답을 서버에게 전송하는 클라이언트로 역할을 나누어 구현하여야 한다. 그러나 온라인상에서는 각기 다른 망 환경이나 호스트 성능 등의 이유로, 동시에 질문이 도달할 수 없으며, 또한 사용자가 작성한 응답이 서버에 도달할 때까지 걸리는

시간도 다를 수밖에 없다. 다음 그림은 온라인 퀴즈 게임에 참여하는 사용자들이 문제를 풀고 응답을 보내는 과정에서 나타나는 지연의 종류와 그들에게 나타날 수 있는 불공정 사례를 보여준다.

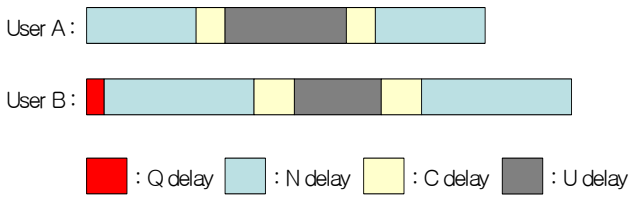

그림 1. 지연의 종류와 불공정 사례

그림에서 보이는 바와 같이 사용자에게 질문을 보내고 응답을 받는데 소요되는 전체 지연은 질문과 답이 망을 통해 전달되는데 소요되는 망 지연과, 질문을 사용자 화면에 보여주고 사용자로부터 답을 입력받는 연산을 수행하는데 소요되는 연산 지연, 그리고 실제 사용자가 질문을 보고 답을 생각해내는데 소요되는 사용자 지연으로 나눌 수 있다. 이때 망 지연과 연산 지연은 각 사용자가 스스로 조정할 수 없는 시스템 지연으로 볼 수 있다. 반면에 사용자 지연은 각 참여자의 경험이나 능력에 따라 달라질 수 있다.

그림에서는 두 사용자에 대한 전체 지연을 표현하고 있는데, 사용자 A가 사용자 B보다도 답을 생각하기 위한 시간을 더 많이 소비해서 사용자 지연이 더 많음에도 불구하고, 망 지연이나 연산 지연이 짧아 전체 지연이 더 작게 나타나고 있다. 이때 단순히 답이 도착한 순서에 따라 승자를 결정한다면 오히려 사용자 지연이 더 많았던 사용자 A가 승자가 되는 불공정한 사례가 발생한다. 즉, 망 지연과 연산 지연의 합인 시스템 지연이 작은 사용자에게 훨씬 유리한 게임 진행이 될 수밖에 없다.

또한 사용자 A와 사용자 B의 처음 부분을 살펴보면 둘이 동시에 시작하는 것이 아니라 시간 차이를 두고 시작한다는 사실도 주목할 필요가 있다. 멀티캐스트 환경이 아닌 유니캐스트 환경의 경우, 서버는 각 클라이언트에게 순차적으로 질문을 보낼 수밖에 없으며, 이에 따라 서버의 전송 큐에서 대기 시간이 서로 다르기 때문에 나타나는 현상이다. 이러한 방식으로 전국에 걸친 참여자들을 대상으로 퀴즈 게임을 진행하고자 한다면 시스템 지연으로 인한 불공정성은 심각한 문제를 야기할 수 있다.

따라서 공정한 게임이 되기 위해서는 전송 순서에 따른 큐잉 지연과 망 환경과 연산 환경에 따른 지연들을 배제하고 순수한 사용자 지연만을 가지고 순서를 결정할 수 있는 알고리즘이 필요하다. 또한 망 지연과 연산 지연을 고려하여 서버의 마감 시간을 설정하는 알고리즘과 이러한 시스템 지연에 따라 전송 순서를 달리하여 전체 지연을 줄이는 알고리즘의 도입이 필요하다.

3. 불공정성을 제거하기 위한 알고리즘

3.1 전공순서결정

다음 그림은 전송 순서를 결정하기 위한 메시지 교환을 보여준다.

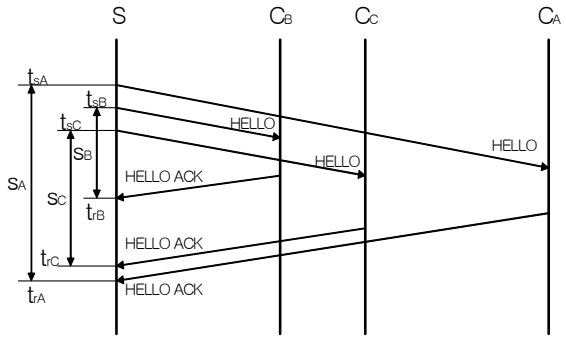

그림 2. 전송 순서 결정을 위한 HELLO 메시지 교환

퀴즈 게임에 참여하고자 하는 사용자들이 들어오면, 서버에서는 임의의 순서로 HELLO 메시지를 보내서 그것에 대한 응답을 요구한다. 각 클라이언트에서는 수신한 HELLO 메시지를 화면에 출력한 후, 응답 메시지를 생성하여 전송한다. 메시지를 수신하자마자 바로 응답 메시지를 보내지 않고 화면에 출력한 후 메시지를 전송하는 것은 망 지연뿐만 아니라 연산 지연도 반영시키기 위함이다. 이때 서버에서는 각 메시지의 전송 시점의 시각과 응답의 도착 시점의 시각을 확인하여 각 사용자의 초기 시스템 지연 값을 설정한다.

모든 사용자의 시스템 지연 값이 수집되면, 서버에서는 그것을 바탕으로 질문 메시지를 전송할 순서를 결정한다. 이때 시스템 지연 값이 가장 큰 사용자에게 제일 먼저 질문을 전송하고, 시스템 지연 값이 가장 작은 사용자에게 제일 늦게 질문을 전송한다. 앞의 그림에서 시스템 지연 값의 크기 순서는 A, C, B이므로 질문 전송 순서도 A, C, B가 된다. 다음 그림은 조정된 전송 순서와 각 사용자들의 지연 구성을 보여준다.

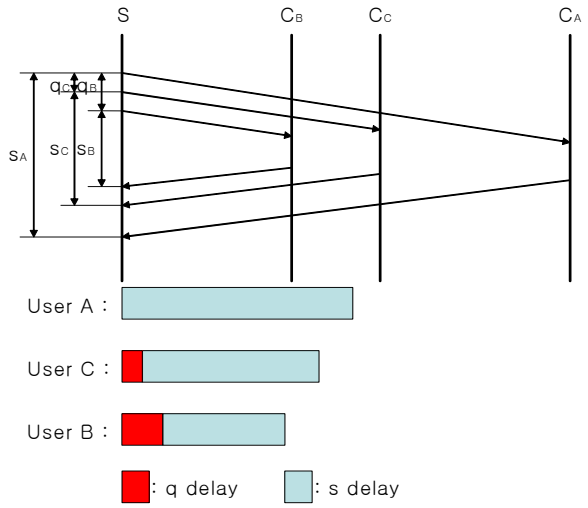

그림 3. 조정된 전송 순서와 지연

3.2 마감시간 결정

원만한 게임 진행을 위한 마감시간 설정이 필요한데, 마감시간의 설정은 클라이언트 측과 서버 측에서 모두 이루어질 필요가 있다. 우선 클라이언트 측에서의 마감시간 설정은 원만한 게임 진행을 위해 필요하다. 클라이언트에서는 질문을 화면에 보인 후 일정 시간동안 대기하면서 사용자의 입력을 기다리고, 주어진 시간 내에 입력이 없는 경우 응답이 없다는 사실을 서버에게 통보한다. 이러한 클라이언트 측 마감시간은 질문을 생성한 서버 측에서 질문마다 임의로 설정하여 동일한 값을 전체 클라이언트에게 전송하고 클라이언트는 이를 바탕으로 자신의 마감시간을 설정한다.

서버 측의 마감시간 설정은 클라이언트로부터 들어오는 응답들을 수집하고 분석하는 시간의 한계를 설정하기 위하여 필요하다. 일부 호스트나 망의 장애로 인해 일부 응답이 전달되지 못하는 상황이 발생할 수 있는데, 이를 대비하기 위해 마감시간을 설정하고, 마감시간 내에 들어온 응답들만을 가지고 게임을 진행한다.

서버 측의 마감시간은 다음 식과 같이 설정된다. 여기서 t_0은 첫 번째 질문을 담고 있는 메시지가 생성되는 시점의 시간이다.

$$timeout_{server} = t_0 + max(q_i + s_i + timeout_{client}) \quad (1)$$

연산의 편의를 위해 마지막 질문을 보낸 시점에서 고려해 보면, 큐잉 지연은 이미 반영된 것으로 볼 수 있으므로 무시할 수 있고, 따라서 다음과 같이 표현할 수 있다.

$$timeout_{server} \simeq t_{last} + timeout_{client} + max(s_i) \quad (2)$$

3.3 승자 판정

게임의 승자 판정은 앞서 언급한 서버 측의 마감시간 내에 들어온 퀴즈 응답들을 이용해 수행한다. 이때 오프라인의 퀴즈 게임과 다른 진행 방식이 필요하다. 오프라인 퀴즈 게임에서는 문제에 신속히 반응하여 정답을 먼저 답한 사용자가 승자가 되는데, 이를 온라인으로 구현하다보면 앞서 지적한 시스템 지연으로 인해 문제에, 신속히 반응하여 보다 짧은 시간에 정답을 입력한 사용자의 응답이 오히려 늦게 도착할 수 있다. 따라서 처음에 도착한 응답뿐만 아니라 마감시간 내에 들어온 모든 정답들의 사용자 지연을 확인하여 그중 사용자 지연이 제일 작은 정답을 보낸 사용자를 승자로 판정해야 한다. 판정이 끝나면 STATUS 메시지로 그 결과를 모든 클라이언트에게 전송한다.

3.4 전송순서 재결정 여부 판단

앞서 언급한 전송순서 결정 알고리즘은 시스템 지연을 기준으로 시스템 지연 값이 큰 클라이언트에게 먼저 메시지를 보내고, 시스템 지연 값이 작은 클라이언트에게는 나중에 보내는 방식으로 전송 순서를 결정하였다. 시스템 지연 값은 망이나 호스트의 상태에 따라 달라질 수 있으므로, 퀴즈 게임이 진행되는 과정에서도 계속 관찰할 필요가 있다. 다음 그림은 퀴즈 질문과 응답이 교환되는 과정에서 지연의 구성을 보여준다.

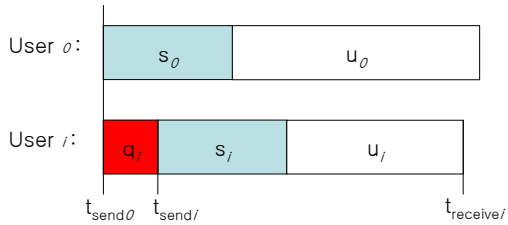

그림 5. 퀴즈 질문/응답 교환시 지연 구성

여기서 s_i를 구하기 위해서는 다음 식을 이용한다.

$$s_i = t_{receivei} - t_{sendi} - u_i \quad (3)$$

따라서 각 응답이 도착한 시간에서 질문을 보낸 시간과 사용자 지연을 감한 값을 이용하여 각 클라이언트의 시스템 지연을 구하고 응답들로부터 구한 순서와 원래 순서를 비교하여 재결정 여부를 판단할 수 있다.

이러한 전송 순서 재결정 여부를 판단하기 위해서는 질문을 전송할 때의 시간과 응답을 받았을 때의 시간을 저장하고, 각 클라이언트의 시스템 지연을 구한 후 이를 다시 정렬시키는 연산을 수행하여야 하므로, 이러한 알고리즘을 매번 수행한다는 것은 전체 성능의 저하를 가져올 수 있다. 따라서 마감시간 내에 모든 응답을 수집하여 정상적인 게임 진행이 가능한 경우에는 기존의 순서를 그대로 유지하고, 마감시간 내에 모든 응답을 받지 못한 경우, 이러한 알고리즘을 수행하여 순서를 재설정하여야 하는지, 아니면 마감시간 자체를 증가시켜야 하는지를 판단하는 방법도 고려할 수 있다.

4. 게임 진행 시나리오

다음 그림은 서버 측의 게임 진행 과정을 보여준다.

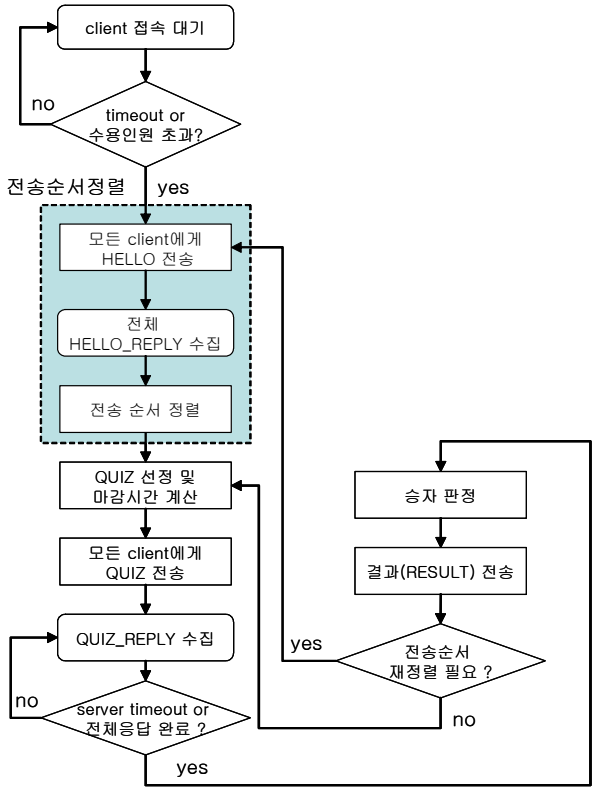

그림 7. 서버 측 게임 진행 알고리즘

서버 측에서 수행할 첫 단계는 일정 시간 혹은 일정 수의 클라이언트들이 접속 할 때까지 대기하는 것이다. 게임에서 정한 일정한 수의 클라이언트가 접속하는 경우, 접속을 마감하고 다음 전송 순서를 결정하는 단계로 진행하게 된다. 퀴즈 게임에서는 출제할 문제를 선정하고, 선정된 문제와 문제마다 미리 정의된 마감 시간을 게임에 참여하는 모든 클라이언트들에게 전송한다.

모든 질문이 전송된 후 응답을 수집하는 단계로 넘어가는데, 계산된 서버 측 마감시간에 도달하거나 모든 클라이언트로부터 응답이 오는 경우 승자를 판정하는 단계로 넘어간다. 승자 판정 단계에서 승자가 결정되면 이를 모든 클라이언트에게 알리고, 그 과정 중에 전송 순서를 달리해야 하는지의 여부를 판단하여 같은 순서를 유지하거나 다시 전송순서를 결정하는 단계를 거쳐 게임을 계속 진행한다.

다음은 클라이언트 측 게임 진행 과정을 보여준다. 서버 측에 비해 진행 단계는 단순하며, 사용자의 구미에 맞는 화면 출력과 사용자 인터페이스의 구현이 더 중요할 수 있다.

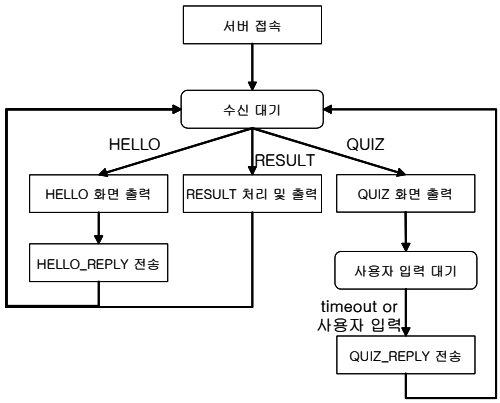

그림 8. 클라이언트 측 게임 진행 알고리즘

5. 결론

본 논문에서는 온라인 퀴즈쇼와 같은 응용에서 망이나 시스템 환경의 차이로 인한 불공정성을 제거하기 위한 진행 시나리오를 제시하였다. 제안한 시나리오에서는 응답이 도착한 순서가 아닌, 질문에 대한 응답을 만들어낸 소요 시간을 기준으로 승자를 판정하도록 하여 망 환경과 연산 환경의 차이로 인한 불공정성을 제거한다. 이러한 게임 진행 시나리오가 적용되면 온라인상에서 다수의 사용자가 참여하는 공정한 퀴즈 게임 프로그램의 구현이 가능 할 것이다.

참고문헌

[1] C. Fidge, "Logical Time in Distributed Computing Systems", Distributed Computing Systems, IEEE Computer Society Press, 1994, pp.73-82

[2] L. Lamport, "Time, Clocks, and the Ordering of Events in a Distributed System", Comm. ACM, Vol.21, No.7, July 1989, pp.558-565