라운드 로빈 실시간 네트워크에서의 가변 길이 트래픽을 위한 오류제어 기법

(An Error Control Scheme for Variable Length Traffic on Round Robin Style Real-Time Networks)

이 정 훈 †

(Jung-Hoon Lee)

요 약 본 논문은 각 노드들이 라운드 로빈 형태로 매체에 접근하는 전송제어 기반 실시간 네트워크에서 시간 제약조건을 고려한 오류 제어 기법을 제안하고 성능을 평가한다. 전송제어 네트워크에는 FDDI, TDMA 및 무선 LAN 등이 속하는데 이러한 네트워크들은 실시간 전송 보장을 위해 오프라인 시에 대역폭 할당을 수행한다. 수신자는 비동기 대역폭을 통해 재전송 요청을 하며 송신자는 대역폭 할당 과정에서 불가피하게 발생한 여분의 대역폭을 이용하여 재전송을 수행하기 때문에 다른 메시지의 전송에 영향을 주지 않고 종료시한 내에서 오류제어가 수행된다. 또 메시지 프레임들로 하여금 메시지의 길이와 순서번호를 포함하도록 하여 메시지의 길이가 주기마다 다른 경우에도 수신자가 빠르게 오류를 보고함으로써 보다 많은 오류에 대한 복구가 가능하다. 분석 결과와 SMPL을 이용한 실험 결과는 제안된 방식이 실시간 통신을 위한 오류제어 기능으로서 네트워크 오류를 극복하여 메시지의 종료시한 만족도를 증가시킬 수 있으며 이중화 혹은 중복 전송 네트워크에 비견할 만한 성능을 보임으로써 적은 비용으로 실시간 네트워크를 구축할 수 있음을 보인다.

키워드 : 실시간 통신, 라운드 로빈 네트워크, 국지 오류제어, 종료시한 만족도

Abstract This paper proposes and analyzes an error control scheme for the transmission control based real-time communication, such as FDDI, TDMA, and wireless LAN, which delivers the message according to the round robin fashion after the off-line bandwidth allocation. Taking into account the time constraint of each message, the proposed error control scheme makes the receiver transmit the error report via asynchronous traffic while the sender resend the requested message via overallocated access time which is inevitably introduced by the bandwidth allocation procedure for hard real-time guarantee. The error control procedure does not interfere other real-time message transmissions. In addition, as each frame contains the size of the message it belongs, the receiver can recognize the end of completion of message transmission. This enables earlier error report to the receiver so that the sender can cope with more network errors. The analysis results along with simulation performed via SMPL show that the proposed scheme is able to enhance the deadline meet ratio of messages by overcoming the network errors. Using the proposed error control scheme, the hard real-time network can be built at cost lower than, but performance comparable to the expensive dual link network.

Key words : real-time communication, round-robin network, local error control, deadline meet ratio

1. 서 론

실시간 응용들이 생성하는 실시간 메시지들은 응용이 요구하는 종료시한(deadline) 이내에 전송이 완료되어야 한다는 시간 제약조건(time constraint)을 가지며 종료시한 이후에 전송이 완료된 메시지들은 손실로 처리된다[1]. 영상 혹은 음성과 같은 실시간 메시지들은 연속적으로 발생되는 스트림의 형태를 보이며 주기 및 전송시간 등의 시간 특성을 갖는데 일반적으로 전송에 있어서 일부 메시지의 손실은 허용한다. 실시간 네트워크는 메시지들의 시간 제약조건을 만족시키기 위해서 오프라

† 정 회 원 : 제주대학교 전산통계학과 교수
jhlee@venus1.cheju.ac.kr
논문접수 : 2001년 10월 4일
심사완료 : 2002년 4월 9일

인 시나 혹은 새로운 실시간 연결이 설정될 때 주어진 메시지 스트림의 집합에 대해 시간 제약조건을 분석하여 메시지 전송 스케줄을 결정하고 이 스케줄에 따라 온라인 시에 전송이 수행되도록 한다. 이 스케줄 및 전송 과정은 네트워크 프로토콜 계층구조상 데이타 링크 계층의 기능에 속하며 보장된 스트림들의 메시지 전송에 대해서는 네트워크에 오류가 발생하지 않는 한 시간 제약조건을 항상 만족시킨다.

다중접근 네트워크에서 사용되는 실시간 통신 방식은 크게 접근중재(access arbitration) 방식과 전송제어(transmission control) 방식의 두 범주로 분류되는데 접근중재 방식은 각 노드가 전송할 수 있는 시점을 결정하는데 중점을 두고 있는 반면, 전송제어 방식은 각 노드가 라운드 로빈 형태로 전송을 하되 노드가 매체를 통해 전송할 수 있는 시간 구간의 길이를 결정하는데 중점을 두고 있다[2]. 즉 접근중재 방식은 공유된 네트워크에 대한 접근 요구가 발생했을 때 이를 우선순위(priority)나 여유시간(laxity)에 준하여 해결하는 방식으로서 우선순위에 기반한 비율단조(rate monotonic) 스케줄링 방식이 이에 속한다. 이와는 대조적으로 전송제어 방식은 각 노드가 네트워크에 배타적으로 접근할 수 있는 시간의 길이를 메시지 스트림의 트래픽 특성에 따라 결정하여 메시지의 시간 제약조건을 만족시키는 방식으로서 FDDI(Fiber Distributed Data Interface)의 시간제약 토큰(Timed Token) 프로토콜이 전형적인 예이다[3]. 더욱이 CSMA/CA에 기반한 무선 LAN에서 실시간 트래픽이 전송되는 CFP(Collision Free Period)에 각 노드가 폴링을 받을 때 매체에 접근할 수 있는 시간을 스트림의 특성에 따라 결정하도록 한다면 무선 LAN도 전송제어 방식에 속한다[4].

실시간 통신 프로토콜들이 메시지들의 시간 제약조건을 만족시킬 수는 있지만 메시지 전송 중의 오류는 불가피하게 발생하며 전송 오류는 메시지의 손실을 의미한다. 지역망 구조에서 전송 오류는 기본적인 매체 오류율, 외부 자극에 의한 일시적인 교란, 혹은 수신자의 버퍼용량 초과 등의 요인에 의해 발생하며 통신 기술의 발달로 인해 매체 오류율이 감소하고 있기는 하지만 완전히 제거되지는 못하고 있다[5]. 또 멀티미디어 응용의 도입은 메시지의 크기를 증가시켜 매체 오류율의 영향을 증폭시키고 있다. 프로토콜 계층 구조상 논리 링크 제어(LLC: Logical Link Control) 계층이 오류를 탐지하고 제어하는 기능을 수행하는데 go-back-N, 정지대기(stop-and-wait), 선택적 재전송(selective repeat) 등 기존의 고전적인 오류제어 기능은 그 과정이 메시지 전송 시간을 연장하게 되어 실시간 통신에는 적합하지 못하다[6]. 이상의 방식들은 확인 혹은 재전송 요청 메시지의 전송과 이에 따르는 메시지의 재전송 과정이 메시지 전송 시간을 연장하여 메시지의 종료시한을 초과하게 할 수 있을 뿐 아니라 재전송되는 메시지가 다른 실시간 메시지 전송을 지연하여 역시 종료시한 초과를 유발할 수 있다.

실시간 통신에서는 대역폭 할당 기법이 최악의 경우를 고려하여 실시간 메시지를 위한 용량 벡터를 결정하기 때문에 각 스트림은 여분의 대역폭을 갖게 되며, 실시간 스트림에 할당하고 남은 대역폭은 비실시간 메시지들이 이용하게 된다. 특정 실시간 메시지 스트림에 있어서 과할당된 대역폭을 이용한 재전송은 종료시한 이내에 수행이 가능할 뿐 아니라 자신에게 배타적으로 할당된 대역폭을 사용하므로 다른 메시지의 전송에 전혀 영향을 주지 않는다. 또 수신자가 송신자에게 전송하는 재전송의 요청은 대역폭을 사전에 확보하기 어려우므로 비실시간 대역폭을 통해 전송하여 수행하여 실시간 스트림의 전송에 주는 영향을 배제한다. 이와 같이 확장된 MAC 기능은 비록 메시지 전송 중에 발생하는 모든 오류를 복구하지는 못하지만 종료시한을 고려하여 오류제어를 수행하고 상위 계층에게 보다 나은 메시지 전송률을 제공할 수 있다.

본 논문은 전송제어 실시간 네트워크, 즉 라운드 로빈 형태를 따르는 경성 실시간 통신에서 효율적으로 동작하는 오류 제어 기법을 제안하고 평가함을 목적으로 한다. 한 네트워크 내에서 다른 실시간 메시지의 전송에 영향을 주지 않고 오류에 의한 종료시한 초과를 최소화하기 위하여 비실시간 대역폭을 통해 오류를 보고하고 과할당된 대역폭을 통해 메시지의 재전송을 수행한다. 본 논문은 다음과 같이 구성된다. 2장에서는 관련연구로서 기존의 실시간 통신을 위한 오류제어 기법과 대역폭 할당 기법을 소개한다. 이를 바탕으로 3장에서는 본 논문에서 제안하는 실시간 오류제어 기법의 기능과 동작을 자세히 기술하고 4장에서는 제안된 기법을 분석함과 아울러 모의실험 결과를 보이고 마지막으로 5장에서는 본 논문을 요약하고 결론과 추후 연구과제를 도출한다.

2. 관련연구

2.1 오류제어 기법

실시간 데이타 링크 계층에서는 두 종류의 오류제어 기법이 널리 사용되는데 이는 시간적 여분(temporal redundancy)과 공간적 여분(spatial redundancy)을 이용하는 방식이다[5]. 시간적 여분을 이용하는 방식은 메

시지의 종료시한이 비교적 긴 분산 응용에 적합한데 이 방식은 물리적인 네트워크의 중복이 없이 동일한 네트워크 상에서 메시지를 중복 전송하여 오류율을 줄이거나 재전송 요청 혹은 타임아웃에 뒤따르는 재전송에 의해 오류를 제어한다. 재전송 요청이나 타임아웃을 이용한 방식은 재전송에 관련된 메시지들이 하부 네트워크 프로토콜에 맞추어 스케줄되어야 한다는 문제점을 갖고 있으며 중복 전송에 의한 방식은 대역폭의 낭비를 크게 하는 문제점을 갖고 있다. 시간적 여분 방식의 예로서 MARS(MAintainable Real-time System)를 들 수 있는데, 이 시스템에서는 그림 1에서 보는 바와 같이 오류 발생 여부에 관계없이 모든 메시지들이 이중으로 중복되어 전송된다[7].

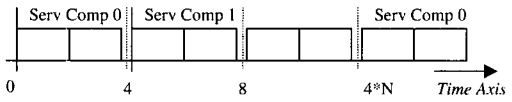

그림 1 MARS의 중복전송 방식

시간적 여분을 이용한 방식의 예로서 2 단계 적응적 오류 기법(two-step adaptive error recovery scheme)은 송신자와 수신자간에 기본적으로 정지대기 방식, 즉 수신자는 하나의 패킷을 받을 때마다 송신자에게 확인 메시지(Acknowledgement)를 보내는 방식을 따른다[8]. 송신자는 재전송 요청을 받았을 때 여유시간 내에 전송 가능한 크기의 오류 보정 코드를 포함한 패킷을 재전송한다. 이 방식은 Reed-Solomon 코드와 같이 여분 문자의 개수가 증가할수록 오류복구의 가능성이 높아지는 코딩 방식에 기반하고 있으나[9] 재전송 요청과정이 정지 대기 방식을 따르고 있기 때문에 오류가 발생하지 않는 경우에도 송신자와 수신자간에 오류제어 메시지가 교환되어야 한다는 단점을 갖고 있다.

이와 아울러 FDDI 네트워크에서 실시간 메시지 전송의 오류를 복구하기 위한 방법으로서 수신자가 연결 설정 과정에서 사전에 결정된 메시지의 크기를 기반으로 오류를 보고하고 과할당된 대역폭을 이용하여 재전송을 수행하는 방식이 제안된 바 있다[10]. 이 방식은 메시지에 포함된 순서 번호와 자신이 토큰을 받는 회수를 세어 오류 보고를 하고 한 주기 내에서 메시지의 전송이 완료된 후 여분으로 토큰을 받는 경우에만 재전송을 수행하여 다른 실시간 트래픽에 대한 영향을 완전히 배제한다. 이 방식은 효율적으로 실시간 메시지의 오류를 복구할 수 있기는 하지만 한 스트림에 속한 메시지들의 길이가 가변적인 경우에는 오류 보고가 늦어질 수 있으며 성능을 개선할 여지가 남아 있다.

반면 공간적 여분을 사용한 방식에서는 중복된 메시지들이 서로 다른 채널이나 네트워크를 통해 전송되며, 메시지의 종료시한이 촉박하여 재전송이나 중복전송을 허용할 수 없을 경우에 적합하다[11]. 이 방식은 항상 메시지들이 다중으로 중복되어 전송되므로 메시지들에 의해 요구된 대역폭의 중복 배수 만큼의 대역폭을 필요로 하게 되어 상당한 대역폭의 낭비를 초래할 뿐 아니라 여러 경로로 다른 시간대에 수신되는 메시지들을 처리하여야 하므로 수신자의 기능이 복잡해진다.

이상의 기법들 이외에 오류 보정 코드에 의한 전향 오류제어 방식이 효율적으로 보이지만 정보의 낭비가 크게 되며 처리시간이 길어지게 된다. 그 예로 Hamming 코드 방식에서는 메시지의 크기를 m 비트라 할 때 한 비트의 오류를 보정하기 위한 오버헤드 k는 $2^k \geq m+k+1$의 관계식을 만족시켜야 되는데 메시지의 크기가 커질수록 대역폭 낭비 요소인 k의 길이가 증가한다[12]. 이 경우에도 2 비트의 동시 오류는 탐지만 가능하며 보정은 할 수 없다. 또 오류 보정 코드는 오류가 발생하지 않는 경우에도 고정적으로 부가적인 정보가 추가되므로 대역폭의 낭비를 초래하며 송신자와 수신자 측에서 오류를 보정하기 위해 많은 데이타 연산을 수행하여야 한다. 오류보정 코드는 메시지의 오류율을 감소시킬 수 있으며 상위계층에서의 오류제어 기능과 결합한다면 오류처리율을 개선할 수 있다.

2.2 대역폭 할당

실시간 메시지들은 주기적으로 발생되며 이들은 주기 내에 전송이 완료되어야 한다. 라운드 로빈 네트워크에서 메시지들의 시간 제약조건을 만족시키기 위해서는 최대 접근 주기나 용량 벡터(capacity vector)와 같은 네트워크 인자들이 정확하게 설정되어야 한다[13]. 최대 접근 주기는 FDDI 프로토콜에서는 TTRT(Target Token Rotation Time)에 해당되는데 각 노드에 있어서 연속된 두 접근의 최대값으로서 주기 내에서 가용한 네트워크 시간을 결정하는 중요한 요소가 된다. 무선 LAN에서도 조정자가 라운드 로빈 방식으로 폴링할 때 한 노드가 폴링을 받는 최대 간격이 된다. 최대 접근 주기가 결정되면 각 노드들은 자신의 메시지 특성에 따라 주기 중 일부를 할당받는데 이 할당받는 양이 용량벡터이다. 각 노드는 토큰을 받거나 폴링된 경우 최대 용량 벡터만큼 실시간 메시지를 전송할 수 있으며 예상 접근 시간 보다 토큰이 일찍 도착했을 때에만 비실시간 메시지를 전송할 수 있다. 실시간 시스템에 있어서 모든 메시지 스트림에 대한 정보는 오프라인 시에 알려지며 매

시지 스트림 S_i는 주기 P_i와 최대 전송 시간 C_i로 특성화되므로 통신 스케줄러는 주어진 $\{S_i\}$ 집합에 대해 시간 제약조건을 만족시키도록 최대 접근 주기 F와 용량 벡터 $\{H_i\}$를 결정하여야 한다. 즉 Ci 크기의 하나의 메시지는 네트워크 접근 권한을 얻을때 마다 최대 Hi 크기의 프레임으로 나뉘어 전송이 된다. 한 스트림이 네트워크 접근 권한을 얻는 횟수는 각 주기마다 다르게 되며 현재까지 제안된 대역폭 할당 기법들은 모두 경성 실시간 제약조건을 만족시키기 위하여 최소의 전송시간을 갖는 주기를 기반으로 용량 벡터를 할당하기 때문에 각 노드는 충분한 대역폭을 할당받게 된다.

3. 제안한 오류제어 기법

3.1 오류의 탐지 및 보고

오류제어를 위해서는 수신자 노드는 수신된 메시지의 오류를 탐지하고 이를 송신자에게 보고하여야 한다. 링 형의 네트워크에서는 프레임이 송신자에게 되돌아오기 때문에 오류 발생 여부를 판단할 수 있지만 버스형 네트워크에서는 반드시 수신자로부터의 확인 혹은 재전송 요청과 같은 오류보고 메시지에 의해 오류가 탐지될 수 있다. 물론 링 형의 네트워크에서도 수신 버퍼 초과에 기인한 기각과 같이 수신자 내부 원인에 기인한 오류는 부가적인 메시지를 통해 보고하여야 한다. 수신자로부터 생성되는 오류 보고 메시지는 다른 실시간 메시지의 전송에 지연을 주지 않도록 비동기 대역폭을 통해 전송하는 방안이 바람직하며 재전송 요청은 SNR 방식과 같이 재전송 리스트를 필요에 따라 주기적으로 전송함으로써 오류 제어 과정이 다른 메시지의 전송에 주는 영향을 최소화할 수 있다[14].

라운드 로빈 네트워크에서 만약 각 주기 중에 전송되는 프레임의 개수를 수신자가 알 수 있다면 오류 보고 시점은 쉽게 결정될 수 있는데 이는 수신자 역시 주기적으로 네트워크에 접근할 권한을 얻기 때문이다. 즉, 수신자 역시 송신자와 같은 회수의 네트워크 접근 권한을 가지므로 접근 횟수를 세고 있으면 송신자가 한 메시지의 전송을 완료했는지 알 수 있다. 그림 2에서는 그 예를 보이고 있는데 한 메시지가 u 개의 프레임으로 나뉘어 전송된다면 수신자는 자신이 네트워크에 접근한 수에 의해 송신자의 전송이 완료되었는지 판단할 수 있다. 처음으로 수신된 프레임이 2라고 한다면 카운터를 2로 설정하고(1번 프레임은 재전송 리스트에 추가) 이후부터 네크워크 접근 회수를 측정하여 최종적으로 u가 된다면 재전송 리스트를 전송한다. 이 방식에 의해 마지막 프레임을 포함한 임의의 프레임에 손상이 발생하더라도 수신자는 쉽게 재전송 시점을 결정할 수 있다. 물론 한 메시지에 속한 모든 프레임이 손상되면 재전송 요구를 할 수 없는데 이 경우는 어차피 종료시한 이내에 오류 복구를 할 가능성이 거의 없다. 재전송되는 프레임의 경우 임의의 순서번호를 갖기 때문에 최초 수신 프레임 번호 인식에 오류를 발생시킬 수 있으므로 일반적인 전송과 재전송되는 프레임을 구분하기 위해 MAC 프레임에 포함된 예약 비트를 사용한다.

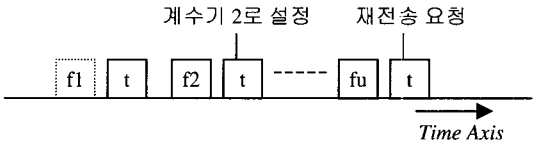

그림 2 프레임 오류의 탐지 및 보고

오프라인 시 혹은 연결 설정 과정에 의해 수신자는 메시지의 최대 크기를 사전에 알 수 있으나 압축 알고리즘과 결합된 멀티미디어 스트림은 매 주기마다 전송되는 메시지의 크기가 가변적이다. 만약 수신자가 최대 메시지 크기만을 알고있다면 수신자는 최대 크기만큼 기다린 다음에야 오류 보고를 할 수 있다. 이 경우 송신자가 전송을 완료한 후 몇 번의 접근 시간이 지난 후에야 오류 보고를 받기 때문에 복구 가능한 오류를 복구하지 못할 가능성이 있다. 즉 [10]에서 제안된 기법은 연결설정 과정에 의해 수신자가 메시지의 최대 크기, 즉 프레임의 개수를 알고 있어서 오류보고가 각 주기마다 최대 프레임 개수만큼 기다린 후에야 오류 리스트를 전송하므로 그만큼 송신자가 재전송할 시간이 줄어든다. 이를 극복하기 위해서는 수신자가 현재 전송되는 메시지의 크기를 알게 하여 현재 주기의 메시지 전송 완료 시점을 판단하도록 하여야 한다. 따라서 메시지를 구성하는 모든 프레임에 현재 전송되는 메시지의 크기 혹은 프레임의 개수를 포함하도록 하면 수신자는 하나의 프레임이라도 받았을 때 메시지의 크기를 알 수 있게 되며 빠른 보고를 통해 송신자는 보다 많은 재전송을 수행할 수 있다. 프레임의 개수에 대한 정보는 역시 프레임의 예약 정보 필드에 넣어 오버헤드를 최소화할 수 있다.

3.2 재전송 기법

수신자 노드는 오류 때문에 순서에 어긋나게 수신된 노드를 재조립할 수 있어야 하며 이를 위해 C_i 크기의 버퍼가 필요하다. 송신자 역시 추후의 재전송을 위해 C_i 크기의 버퍼를 할당받는데 재전송 가능성은 주기 이내

에서만 발생 가능하며 일반적으로 송신 버퍼는 새로운 메시지가 생성될 때마다 자신의 버퍼에 덮어쓰게 된다. 또 다른 오류 제어 기법과 마찬가지로 각 프레임마다 순서번호를 추가하며 수신자는 재전송에 의한 프레임 수신 순서의 전도를 복구할 수 있다고 가정한다.

그림 3은 송신자와 수신자의 오류 제어 알고리즘 동작을 보여주고 있다. 송신자는 그림 3(a)에서 보는 바와 같이 하나의 메시지를 보내는 과정에서 Normal, Retrans, Other 등 3 단계로 진행하는데 Normal 상태에서는 토큰을 맞이할 때마다 한 프레임씩 전송을 하며 이 상태에서 모든 프레임을 전송할 수 있도록 대역폭 할당이 되어 있다. 모든 프레임의 전송이 완료된 후 수신자로부터의 오류 보고가 도착하면 Retrans 단계로 상태가 변경되는 반면 오류 없는 전송이 보고되거나 오류 보고가 도착하지 않으면 Others 상태로 천이하여 새로운 메시지가 도착할 때까지 비실시간 메시지를 전송한다. Retrans 단계에서는 토큰을 맞이할 때마다 오류 리스트에 포함된 프레임들을 재전송한다. 이 과정에서 새로운 메시지가 도착하면 재전송 과정을 기각하여야 하며 재전송이 완료되면 다시 수신자로부터의 오류보고를 기다린다. 그림 3(b)는 수신자의 동작을 보여주고 있는데 수신자는 하나의 메시지를 받기 위해 Normal, Retrans 상태를 갖는다. 첫 프레임을 수신하여 카운터를 설정한 후에는 토큰을 받을 때마다 카운터를 감소시키며 카운터가 0으로 되면 오류리스트를 송신자에게 전송한다. 이후 Retrans 상태로 천이하여 재전송되는 프레임들을 수신하며 이 과정에서 새로운 메시지의 첫 프레임이 도착하면 다시 Normal 상태로 천이하여 카운터 감소작업을 계속한다.

4. 분석 및 모의 실험 결과

본 절에서는 오류제어 기능이 없는 단일 네트워크와

```
event new message arrival : state = Normal
switch (state) {
   case Normal :
          event token reception : send each frame
          event completion and error report arrival : state - Retrans
          event completion and no error report : state - Others
   case Retrans :
          event token reception : retransmit a frame on error list
          event new message arrival : state - Normal
          event completion and error report arrival : state - Retrans
          event completion and no error report : state - Others
   case Async :
          event token reception : send asynchronous frame
          event new message arrival : state - Normal
}
```

(a) 송신자 노드의 동작

```
event arrival of the first frame of a new message : state - Normal
(initialize the error list and set counter)
switch (state) {
    case Normal :
          event token recpetion : decrement the counter
          event counter expiration : send error list and state - Retrans
          event frame reception : process the frame
          event frame missing : add the frame number to error list
    case Retrans:
          event arrival of the first frame of a new message : state - Normal
          event frame reception : delete the frame from the error list
          event counter expiration : send error list and state - Retrans
          event frame missing : add the frame number to error list
}
```

(b) 수신자 노드의 동작

그림 3 송수신자 노드의 알고리즘

제안된 오류제어 기능을 갖는 네트워크의 메시지 오류율을 분석 및 모의실험에 의해 측정한다. 제안된 방식의 성능과 비교하기 위한 또다른 기준으로 MARS와 같이 프레임을 중복하여 전송하는 방식을 선택한다. 이 방식에서는 대역폭의 낭비가 심하지만 각 프레임의 중복이 모두 손실되었을 때에만 프레임의 손실로 처리되므로 메시지 오류율이 상대적으로 낮으므로 본 논문에서 제안한 방식과의 비교기준이 될 수 있다.

4.1 분석

2.2에서 소개된 바와 같이 주어진 스트림 집합 $\{S_i\}$의 원소 S_i는 트래픽 특성으로 주기 P_i와 전송시간 C_i를 갖는다. 대역폭 할당 기법은 이를 기반으로 최대 접근 주기 F와 용량벡터 $\{H_i\}$, 즉 노드의 최대 네트워크 접근 시간을 결정한다. 결국 $\{S_i\}$에 대해 메시지의 오류율, 혹은 전송오류에 의한 실기율(deadline miss ratio)은 식 (1)과 같이 정의되는데 F_i와 E_i^X는 각각 메시지 스트림 S_i의 발생 빈도와 오류 발생률이다.

$$E^X = F_i \cdot E_i^X, \quad X\text{는 오류제어 방식} \tag{1}$$

발생 빈도 F_i는 T를 충분히 큰 임의의 시간 구간이라 정의할 때 식 (2)와 같이 계산된다.

$$F_i = \frac{\frac{T}{P_i}}{\sum_{i=1}^{n}\frac{T}{P_i}} = \frac{\frac{1}{P_i}}{\sum_{i=1}^{n}\frac{1}{P_i}} = \frac{1}{P_i \cdot \pi}, \quad \text{단} \quad \pi = \sum_{i=1}^{n}\frac{1}{P_i} \tag{2}$$

ρ를 비트 오류율이라고 할 때 오류제어를 수행하지 않는 단일 네트워크 상에서 스트림 S_i에 오류가 발생할 확률 E_i^s는 잘 알려진 바대로 크기 C_i인 메시지에 오류가 발생할 확률로 계산되므로 식(3)과 같이 계산된다[6].

$$E_i^s = 1-(1-\rho)^{C_i} \cong \rho \cdot C_i \tag{3}$$

이 과정에서 프레임 분할은 오류율에 영향을 주지 않으며 결국 오류제어를 수행하지 않는 단일 네트워크에서의 전체적인 메시지 오류율 E^s는 식(4)와 같이 계산된다.

$$\begin{aligned} E^s &= \sum_{i=1}^{n} F_i \cdot E_i^s = \sum_{i=1}^{n}\frac{1}{\pi P_i} \cdot E_i^s \\ &= \frac{1}{\pi} \cdot \sum_{i=1}^{n}\frac{E_i^s}{P_i} = \frac{1}{\pi} \cdot \sum_{i=1}^{n}\frac{\rho \cdot C_i}{P_i} \\ &= \frac{1}{\pi} \cdot \rho \cdot \sum_{i=1}^{n}\frac{C_i}{P_i} = \frac{1}{\pi} \cdot \rho \cdot U, \text{ 단 } U = \sum_{i=1}^{n}\frac{C_i}{P_i} \end{aligned} \tag{4}$$

제안된 기법의 오류율을 구하기 위해 복구 가능한 오류율을 구하도록 하며 이는 한 프레임의 오류 발생률과 한 주기 내에서 사용가능한 여분의 대역폭에 의한 함수로서 계산된다. 한 프레임의 오류 발생률은 식 (3)에 C_i 대신 H_i를 대입함으로써 $\rho \cdot H_i$ 와 같이 계산되며 한 주기내 여분의 대역폭 R_i는 전체 실시간 트래픽, 비실시간 트래픽 및 오류 처리를 위한 트래픽의 확률 함수에 의해 결정된다. 즉 네트워크 전체적으로 실시간 트래픽이 많다면 여분의 대역폭이 감소하게 되며 또 각 주기가 최대 접근 주기로 약분되지 않는 경우 각 주기마다 맞이하는 여분의 토큰을 맞이하는 횟수가 다를 뿐 아니라 해석적 모델로 표현될 수 없다[3]. 이 과정에서 R_i의 확률 밀도 함수를 구할 수는 없지만 평균값을 구할 수는 있으므로 평균값을 사용하여 근사치를 구하도록 한다. 각 스트림이 주기 내에서 맞이하는 평균 여분 대역폭 R_i는 전체 대역폭에서 실시간 및 비실시간 트래픽이 차지하는 비율을 제거함으로써 식 (5)와 같이 계산된다.

$$R_i = (1-U-\delta) \cdot P_i \tag{5}$$

식(5)에서 δ는 기타 트래픽이 차지하는 비율로서 P_i 내에서 평균 비동기 트래픽과 오류제어 트래픽이 차지하는 비율, 토큰 회전 오버헤드 및 오류를 보고하는데 있어서의 지연시간을 포함한다. 결국 한 주기 내에서 각 스트림은 평균적으로 재전송 과정 중에 발생한 프레임 오류를 포함하여 $\lfloor R_i / H_i \rfloor$ 개 까지의 프레임 오류를 제어할 수 있으며 스트림 S_i에 있어서의 오류율 E_i^P을 계산하기 위해 u를 한 메시지 전송에 필요한 프레임 수, v를 제어가능한 오류발생 프레임 수라 하면 이들은 각각 $\lceil C_i / H_i \rceil$와 $\lfloor R_i / H_i \rfloor$로 계산된다.

만약 k 개의 프레임을 보낼 때 w 개의 여분의 프레임이 있다면 오류없이 전송될 확률 T는 (6)과 같이 재귀적으로 계산된다.

$$\begin{aligned} &T(k,w) = \sum_{i=0}^{m} {}_kC_i \cdot p^i \cdot (1-p)^{k-i} \cdot T(i, w-i), \\ &T(0,w) = 1, \quad T(k,0) = (1-p)^k \end{aligned} \tag{6}$$

식 (6)에서 m은 k와 w 중 작은 값이며 p는 한 프레임의 오류율이다. $T(0, w)$는 전송할 프레임이 없는 경우이므로 모두 1이다. 반면 $T(k, 0)$은 k 개의 프레임을 보냈고 여분의 프레임이 없는 경우에 오류 없이 전송될 확률이므로 모든 프레임이 오류없이 전송되어야 한다. 만약 k 개의 프레임을 보냈고 i 개의 오류가 발생하였다면 i 개는 재전송되었을 것이므로 남은 프레임의 개수는 최소 i 만큼 감소하며 $T(i, w-i)$를 또 계산한다. 결국 k와 w의 작은 값까지의 범위에 있는 i에 대해 발생 확률과 오류 복구 가능성을 곱하면 최종적인 전송성공 확률이 계산된다. 따라서 각 스트림에 대해 u와 v의 평균값을 구한 후 다음의 식 (6)에 대입하면 전송 성공률 $(1 - E_i^P)$의 근사치를 얻을 수 있으므로 E_i^D를 구할 수 있다. 이 식은 모든 오류 보고가 성공적으로 한 접근 주

기 이전에 수행되는 것으로 가정하고 있다.

이중 네트워크를 통한 중복 전송 방식은 모든 프레임이 이중으로 중복되어 전송되므로 두 프레임이 모두 손실되었을 때에만 프레임의 손실이 발생한다. E_i^D는 식 (7)과 같이 계산되며 역시 메시지 오류율 E^D는 식(1)에 E_i^X 대신 E_i^D를 대입함으로써 구해진다.

$$E_i^D = 1-(1-(\rho \cdot H_i)^2)^{\alpha} \tag{7}$$

4.2 도의 실험

제안된 기법과 오류제어가 없는 네트워크, 이중링크에 의한 중복 전송 등의 기법에 대해 매체 오류율, 사용률, 비동기 트래픽 및 메시지의 최대-평균 길이 비율 등의 네트워크 인자에 따라 종료시한 만족도를 SMPL을 이용한 모의실험에 의해 측정하였다[15]. 각 실험은 해당 사용률을 갖는 임의의 스트림 집합을 50 개 생성한 후 다른 인자들의 값을 고정시킨 채 하나의 인자의 값을 변화시키면서 종료시한 만족도를 측정하였다. 실험에서 대역폭 할당은 기존의 연구결과를 이용하여 식 (8)과 같이 결정하였다[3]. 식 (8)에서 P_{min}은 주기의 최소값이며 γ는 한 접근 라운드의 낭비시간이다.

$$F = \frac{P_{\min}}{\frac{-3+\sqrt{9+\frac{8P_{\min}}{2}}}{2}}, \qquad H_i = \frac{C_i \cdot P_i}{U} \cdot (F-\gamma) \tag{8}$$

그림 4~8은 모의실험 결과를 보이고 있는데 각 그림에서 크기보고는 본 논문에서 제안된 오류제어 기법을 의미하며, 최대보고는 [10]에서 제안된 바와 같이 메시지의 최대 길이에 해당하는 프레임을 수신한 후 오류를 보고하는 방식이다. 또 단일링크는 오류제어 기능없이 하나의 링크를 통해 전송하는 방식이며 이중링크는 고비용의 이중화된 링크를 통해 중복 전송하는 방식이다.

그림 4는 매체 오류율과 메시지들의 실기율을 보여주고 있다. 매체 오류율에 따른 종료시한 만족도 측정을 위해 0.70 ~ 0.79까지의 사용률을 갖는 50 개의 각 스트림의 집합에 대해 식 (8)에서 같이 각각 대역폭을 할당한 후 매체 오류율을 10^{-5}에서 10^{-8}까지 변화시키면서 종료시한 만족도를 측정하였다. 실험에서 비동기 트래픽의 부하는 0.2, 최대-평균 메시지의 비율은 1.2로 설정하였다. 그림 4에서 보는 바와 같이 이중 링크를 통해 고비용으로 중복 전송하는 경우가 제일 실기율이 낮으나 본 논문에서 제안하는 저비용의 오류 제어 기법도 거의 비슷한 성능을 보여주고 있으며 단일링크 방식이나 최대보고 방식으로 오류 제어를 수행하는 경우보다 성능을 더욱 개선하였다.

그림 5는 메시지 길이의 최대-평균 비율에 따른 실기율을 보여주고 있는데 각 스트림 집합의 최대 사용률은 0.70에서 0.79의 값을 가지며 비동기 트래픽 부하 0.2, 매체 오류율을 10^{-6}로 설정하고 최대-평균 비율을 1.0부터 2.0 까지 변화시켰다. 모든 기법들의 실기율이 감소하는 이유는 생성된 메시지의 수는 동일하지만 각 메시지의 길이가 감소함에 따라 메시지 오류율도 같이 감소하기 때문이다. 최대-평균 비율이 같은 경우는 최대보고와 크기보고 두 방식이 같은 성능을 보여주지만 비율이 커질수록 그 차이는 크게 나타난다.

그림 6은 비실시간 부하를 0에서 0.3 까지 변화시켜 가면서 실기율을 측정한 결과인데 비실시간 부하가 0인 경우는 최대보고와 크기보고가 거의 유사한 성능을 보여준다. 한 노드가 접근 권한을 얻었으나 보낼 메시지가 없을 때는 다음 노드에게 바로 권한을 넘기는데 이 과정에서는 큰 낭비시간이 발생하지 않는다. 즉, 비실시간 트래픽이 없으므로 오류 보고가 몇 주기 이후에 수행되더라도 송신자는 크기보고 방식과 같은 과할당 대역폭을 갖기 때문이다. 그러나 비실시간 트래픽이 증가하는 경우에는 최대보고 방식은 오류 보고를 늦게 받을 가능성이 높을 뿐 아니라 과할당된 대역폭의 양이 감소하므로 실기율이 단일보고 방식에 근접한다. 반면 크기보고 방식은 조금씩 실기율이 증가하기는 하지만 여분의 대역폭을 이용할 수 있기 때문에 메시지 전송 확률이 높다.

그림 7은 사용률에 따른 실기율을 보여주고 있는데 매체 오류율은 10^{-6}으로 고정하고 스트림 집합의 사용률을 0.5에서 0.8까지 변화시켰으며 비동기 트래픽의 비율은 0.2, 최대-평균 비율은 0.2로 설정하였다. 사용률이 높아질수록 제안된 기법의 실기율이 증가하는 이유는 복구할 수 있는 오류의 수에 직접적으로 영향을 주는 여분의 대역폭 양이 감소하기 때문이다. 사용률이 낮아질수록 모든 프레임마다 이중으로 중복되어 전송할 확률이 높아지므로 중복전송 방식과 성능이 유사하게 된다.

그림 8은 분석 결과에 의한 근사치를 평가하기 위하여 비실시간 트래픽을 0으로 설정하였다. 최대보고와 크기보고 방식의 성능이 유사한데 한 스트림은 전송이 완료된 후 접근 권한을 얻는 경우 바로 권한을 넘기므로 오류 보고나 여분의 대역폭 양에 있어서 두 방식이 오직 네트워크 오버헤드 정도 밖에는 차이가 없으며 이 차이는 전체적인 성능에 큰 영향을 주지 않기 때문이다. 따라서 하나의 그래프로 표현하였다.

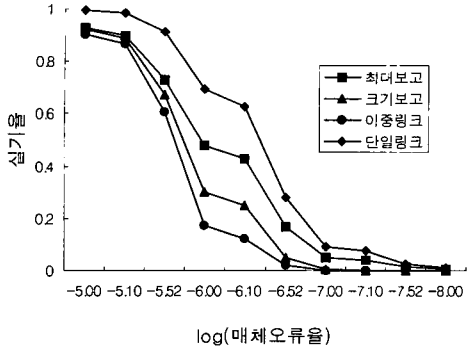

그림 4 매체오류율에 따른 실기율(사용률=0.7, 비동기 트래픽 부하=0.2, 최대-평균비율=1.2)

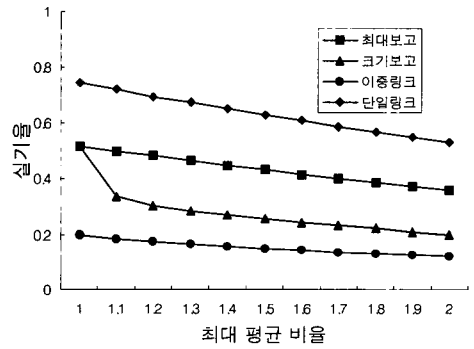

그림 5 최대-평균 비율에 따른 실기율(매체오류율=10^{-6}, 사용률=0.7, 비동기트래픽 부하=0.2)

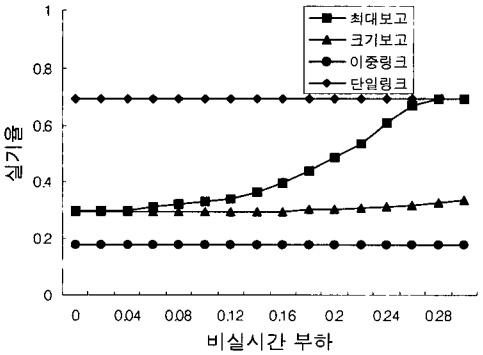

그림 6 비실시간 부하에 따른 실기율(매체오류율=10^{-6}, 사용률=0.7, 최대-평균비율=1.2)

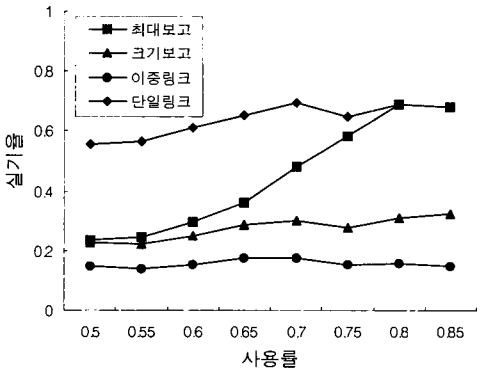

그림 7 사용률에 따른 실기율(매체오류율=10^{-6}, 비동기 트래픽 부하=0.2, 최대-평균비율=1.2)

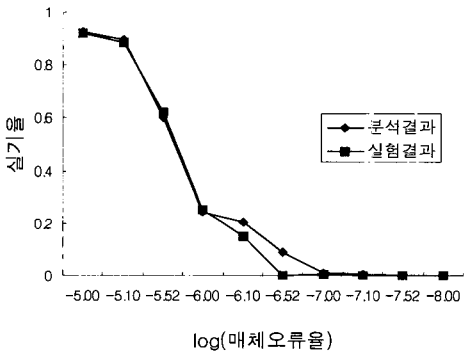

그림 8 분석결과와 실험결과의 비교(비동기 트래픽 부하=0, 사용률=0.7, 최대-평균비율=1.2)

5. 결 론

기존의 연구에 의하면 최근까지 제안된 실시간 통신을 위한 오류제어 기법들이 데이타의 종료시한을 직접적으로 고려하지 않았기 때문에 슬라이딩 윈도우와 같은 고전적인 오류제어 기법에 비해 약간의 개선을 보여줄 뿐 시간 오류에 대한 근본적인 해결을 하고 있지는 않다고 할 수 있다. 결국 MAC 계층에서 메시지의 오류율을 감소시켜 상위 계층에서의 오류율을 감소시키는 방안이 바람직한데 본 논문에서 제안된 방식은 하부 네트워크의 MAC에 기반하여 이의 특성을 이용함으로써 기존 오류제어 방식의 비실시간성을 극복할 수 있다.

제안된 방식은 수신자의 네트워크 접근 횟수를 측정하여 오류 보고 시점을 결정한 후 비동기 대역폭을 통

해 오류를 보고하는 한편 송신자는 과할당된 대역폭을 통해 재전송함으로써 다른 실시간 트래픽의 전송에 전혀 영향을 주지 않는다. 또 주기마다 메시지의 길이가 다른 경우에도 각 프레임에 현재 전송되는 메시지의 길이를 포함하여 전송함으로써 수신자는 빠르게 오류를 보고할 수 있으며 이에 따라 송신자는 더많은 여분의 대역폭을 이용해 보다 많은 프레임 오류를 복구할 수 있다. 매체오류율, 사용율, 최대-평균 비율 등 다양한 인자들에 기반한 모의실험은 제안된 기법이 가변적인 길이를 갖는 메시지 스트림 집합에 대해 종료시한 만족도를 향상시킬 수 있음을 보여준다.

추후 연구과제로서 이중 전송에 의한 오류율 감소보다 실기율을 더욱 감소시키기 위해서는 오류제어 라운드를 여러 번 수행할 수 있도록 대역폭을 할당할 때 추가적으로 미리 확보하여 R_i를 늘이는 방안이 강구될 예정이다. 이와 아울러 본 논문에서는 메시지의 종료시한이 주기와 같다고 가정하였으나 종료시한이 주기보다 큰 경우는 여러 주기간에 오류제어가 수행될 수 있도록 보완되어야 한다.

참 고 문 헌

[1] K. Arvind, K. Ramamritham, J. Stankovic, "A local area network architecture for communication in distributed real-time systems," *Journal of Real-Time Systems*, Vol. 3, pp.115-147, May 1991.

[2] N. Malcolm and W. Zhao, "Hard real-time communication in multiple-access networks," *Journal of Real-Time Systems*, pp.37-77, 1995.

[3] S. Mirchandani, R. Khanna, *FDDI Technology and Applications*, Johm Wiley and Sons, Inc., 1993.

[4] J. Lee, S. Kim, Y. Lee, "A low cost local error control scheme for hard real-time messages on wireless LAN," *IEEE ICWHLN*, pp.69-78, Dec. 2001.

[5] J. Kurose, "Real-time communication on packet-switched networks," *Proc. of IEEE*, Vol. 82, No. 1, pp.122-129. Jan. 1994.

[6] M. Schwartz, *Telecommunication Networks: Protocols, Modeling and Analysis*, Addison Wesley Publishing Company, 1987.

[7] H. Kopetz, A. Damm, J. Reisinger, W. Schwabl, "The real-time operating systems of MARS," *Proc. Operating Systems Review*, pp.141-157, 1988.

[8] D. Qiao, K. Shin, "A two-step adaptive error recovery scheme for video transmission over wireless networks," *IEEE INFOCOM*, pp.1698-1704, Mar 2000.

[9] H. Bengtsson, E. Ulemann, P. Wiberg, "Protocol for wireless real-time systems," *Proc. Euromicro Conference on Real-time Systems*, 1999.

[10] 이정훈, 김호찬, "FDDI 기반 실시간 데이타 수집 네트워크에서의 최선노력 오류제어 기법", 한국정보과학회 논문지: 정보통신, pp.347-354, 2001년 9월.

[11] H. Garcia-Molina, B. Gao, D. Barbara, "Aggressive transmissions over redundant paths," *Proc. Distributed Computing Systems*, pp.198-207, 1991.

[12] Z. Kohavi, *Switching And Finite Automata Theory*, McGraw-Hill, 1978.

[13] B. Chen, G. Agrawal, W. Zhao, "Optimal synchronous capacity allocation for hard real-time communications with the timed token protocol," *Proc. Real-Time Systems Symposium*, pp.198-207, 1992.

[14] A. N. Natravali, W. D. Roome, K. Sabnani, "Design and implementation of a high-speed transport protocol," *IEEE Trans. Communication*, Vol. 38, No. 11, pp.2010-2024, Nov. 1990.

[15] M. H. MacDougall, *Simulating Computer Systems: Techniques and Tools*, MIT Press, 1987.

이 정 훈

1988년 서울대학교 컴퓨터공학과 공학사. 1990년 서울대학교 컴퓨터공학과 석사. 1996년 서울대학교 컴퓨터공학과 박사. 1990년 ~ 1991년, 1996년 대우통신 근무. 1997년 ~ 현재 제주대학교 자연과학대학 전산통계학과(조교수). 관심분야는 실시간 통신, 분산 시스템, 클러스터 컴퓨팅