

멀티캐스트 전송의 신뢰성을 위한 FEC기반 오류제어 메카니즘

정회원 박재영*, 최승권**, 신동화**, 송영옥***, 신승수**, 조용환**

Error Control Mechanism based on FEC for Reliable Multicast Transmission

Jae-Young Park*, Seung-Kwon Choi**, Dong-Hwa Shin**, Young-Ok Song***,
Seung-Soo Shin**, Yong-Hwan Cho** *Regular Members*

요약

통신망의 발달에 따른 데이터의 증가는 어떻게 데이터를 전달하고자 하는 곳까지 효율적으로 데이터의 성질을 변화시키지 않고 오류 없이 전달할 것이냐는 문제를 발생시켰으며, 이를 해결하기 위해 신뢰성 보장에 관한 연구가 진행되고 있다. 이러한 신뢰성에 대한 문제는 UDP 기반의 멀티캐스트에서는 더욱 필요하다 할 수 있다. 하지만 많은 수신자가 존재하는 멀티캐스트에서 기존의 방법으로 접근하였을 때 피드백 폭주 및 대용량의 데이터를 전송할 때 발생하는 지연 등의 문제가 야기 될 수 있다.

본 논문은 데이터 전송 과정에서 네트워크의 오류 및 폭주로 인해 발생할 수 있는 신뢰성 문제를 송신자와 수신자 측에서 자체적으로 복구할 수 있는 차등 FEC를 기반으로 하여 복구 할 수 있도록 하였으며, 시뮬레이션 결과 기존의 방법보다 향상된 결과를 보였다.

ABSTRACT

The increment of data according to advance of network technology causes problems of errorless transmission, and many researches are on going to solve this problem. Especially, it is need multicast based on UDP, but it may causes problems like delay on data transmission and feed-back congestion approached by conventional method that exist many receivers.

This paper proposes an Adaptive FEC(Forward Error Correction) mechanism to solve congestion problem occurred in sending data using multicast, and simulation result shows that the Adaptive FEC is more effective than conventional FEC.

I. 서론

통신망의 발달은 일대일 통신을 뛰어넘는 다자간의 통신을 가능케 하고 있다. 이에 따라 다자간 통신에 사용되는 일대다 통신 기법인 멀티캐스트에 관련된 연구가 활발히 진행되고 있으며 화상회의 시스템, 전자 칠판과 같은 멀티캐스트 응용프로그램들이 등장하고 있다.

수송계층에서의 신뢰성은 주로 전송 품질 저하로 인한 데이터 오류를 복구하는 오류제어기법에 의해 제공된다. 통상 멀티캐스트는 신뢰성을 제공하는 TCP와 다른 신뢰성이 없는 UDP기반이므로 신뢰성을 제공하지 못했다. 하지만 멀티캐스트에서 신뢰성을 제공하는 많은 연구들이 진행되어지고 있다.^{[1][2][3]}

신뢰성 있는 멀티캐스트 전송을 위해서 수송계층에서 오류제어, 흐름제어, 혼잡제어를 고려해 주어

* (주)EMBTEK, ** 충북대학교 컴퓨터공학과(yhcho@cbucc.chungbuk.ac.kr),
논문번호 : 00265-0710, 접수일자 : 2000년 7월 10일

*** (주)TeraBit Tech.

야 한다. 본 논문에서의 주 내용인 오류제어는 수신자가 수신한 데이터에 오류가 있는지를 감지하고 손실이 발생되었을 경우에는 손실을 송신자에게 보고하여 수신자가 전송되는 방법을 적절히 조정해줌으로써 이루어지게 되며, 이 과정에서 많은 수신자가 존재하는 멀티캐스트의 특성상 수신자들로부터의 피드백이 제어할 수 없을 만큼 들어오는 피드백의 폭주 현상이 문제점으로 작용하게 된다. 이러한 문제를 해결하기 위해서 피드백 정보를 분산시키는 방법과 잉여 정보를 삽입하여 전송하는 FEC (Forward Error correction)방법이 있다. 피드백 정보를 분산시키는 방법에는 재전송 방법을 사용하므로 FEC를 이용하는 방법에 비해서 지연시간과 같은 면에서 비효율적이라고 말할 수 있다. 반면 FEC 방식은 데이터의 부호화를 이용하여 수신자 측에서 오류가 발생하였을 경우 그 여분의 데이터를 가지고 자체적으로 에러를 복구함으로써 네트워크에서 발생하는 오류상황에 대해 문제점을 해결하는 해결책으로 작용할 수 있다. 그러나 오류가 적게 발생하는 경우에는 에러 검출과 정정을 위해 부가된 여분의 데이터가 그만큼의 부담으로 작용하게 된다는 문제점이 생기게 된다.

본 논문에서는 네트워크 환경에 보다 나은 신뢰성 있는 멀티캐스트 전송을 위해 오류를 제어하는 방법에서 원칙적으로 재전송하는 ARQ나 고정적인 FEC방법이 아닌 FEC와 ARQ를 계층적으로 차등 적용하여 수신자 측에서 오류가 발생하더라도 자체적으로 복구하게 함으로써 오류가 발생하는 상황에 적절히 대응하도록 한다.

II. 신뢰성 있는 멀티캐스트 오류제어

멀티캐스트의 신뢰성을 제공(RMT : Reliable Multicast Transfer)하기 위해서는 오류제어, 흐름제어 및 폭주제어등의 기술이 지원되어야 한다. 이 중에서 흐름제어와 폭주제어 기술은 모두 송신자의 데이터 전송속도 혹은 전송량을 통제하는 기술로써, 최근에 들어서는 같은 부류의 기술로 취급한다.

오류제어 기술은 수신자의 입장에서 손실(loss)된 멀티캐스트 전송 패킷을 복구(recovery)하는 절차를 의미한다. 좀 더 세부적으로 살펴보면 먼저 수신자가 패킷의 손실 여부를 탐지하는 오류탐지(error detection), 송신자 혹은 다른 수신자에게 손실된 패킷의 재전송을 요구하는 재전송 요구(retransmission request), 그리고 이를 토대로 손실된 패킷을 재전송

(retransmission) 하는 기능으로 나누어 볼 수 있다. TCP와는 달리 멀티캐스트에서 신뢰성 제공이 어려운 이유는 바로 여러 수신자들이 서비스에 참여하고 있다는 점이다. 특히 손실 패킷의 재전송 요구시 수많은 수신자가 동시에 송신자에 재전송 요구 패킷을 보내는 경우, 소위 ACK(acknowledgement) 혹은 NACK(negative ACK) 폭주(implosion)문제가 발생한다. 대규모 그룹 및 네트워크에 쉽게 적용할 수 있는 확장성(scalability)문제를 해결하기 위해 그동안 많은 연구들이 진행되어왔다.

신뢰성 있는 멀티캐스트에서는 송신자가 모든 수신자들의 수신상태 정보를 관리 유지해야한다. 많은 수신자를 갖는 송신자는 모든 수신자들로부터의 상태정보를 바탕으로 재전송을 수행해야 한다. 이 때문에 수신자의 증가에 따라 송신자의 부하가 증가하는 Scalability 문제와 수신자들이 전송하는 수신상태 정보가 송신자 측에서 폭주하는 문제가 발생한다.

이러한 문제, 특히 송신자주도 오류제어 기법의 확장성 문제를 해결하기 위한 기법으로 수신자주도 (receive-initiated), 토큰기반(token-based), 서버기반 (server-based) local recovery, 트리기반(tree-based) local recovery, 응용서비스 기법(application specific) 과 같이 분류할 수 있다.

송신자주도기법에서는 ACK 메시지를 토대로 송신자측에서 오류제어를 수행한다. 그렇지만 수신자 주도기법은 개별수신자가 오류제어의 책임을 진다. 송신자는 수신자로부터 NAK (negative ACK)가 올 때까지 데이터전송을 계속한다. 수신자로부터 NAK 메시지가 오는 경우, 송신자는 해당데이터를 특정수신자에게 재전송 한다. 각수신자는 데이터오류를 탐지한 후, NAK 메시지를 송신자에게 유니캐스트로 전송한다. 이 경우에 여러 수신자가 같은 NAK 메시지를 하나의송신자에게 전송하는 문제가 발생할 수 있다. 이를 NAK implosion이라 한다. NAK방식 자체로서는 흐름제어(flow control) 측면에서 완전한 수송계층 프로토콜이 아니다. 송신자입장에서 데이터전송률(혹은 window size)을 제어하기 위해서는, 어떤 형태로든 ACK 정보를 받아야 하기 때문이다.

토큰기반 오류제어기법은 토큰을 가진 수신자만이 오류제어기법을 수행한다. 이러한 방식의 대표적인 예는 RMP(reliable multicast protocol)[4] 등이 있다. 토큰기반 오류제어방식에서는, 전체수신자그룹이 하나의 링(ring)으로 구성된다. 송신자의 데이터 전송에 대하여 토큰을 가진 수신자만이 송신자에게

ACK 메시지를 보낼 수 있다. ACK 메시지를 보낸 후에, 곧바로 다음 수신자에게 token을 건네준다. 데이터손실을 경험한 수신자는 누구나 NAK를 송신자에게 전달할 수 있으나, 이는 흐름제어의 일환으로 송신자의 window size를 감소시키기 위해 사용될 뿐, 데이터재전송은 해당수신자가 토큰을 얻게 되었을 때 한꺼번에 일어난다. 이처럼 토큰사용자만이 ACK를 전송하고, 손실된 데이터에 대하여 재전송을 할 수 있다. 토큰기반 오류제어기법은 비록 ACK/NAK 등의 제어메시지를 줄일 수는 있으나, 실시간으로 오류복구를 요구하는 서비스에는 적용하기 어렵다.

서버기반 local recovery 방식에서는, 미리 선정된 하나 혹은 여러 개의 서버가 오류제어 및 데이터재전송을 담당한다. 따라서 송신자의처리부담을 여러 개의 서버로 분산시킬 수 있다. 이러한 방식을 사용하는 전형적인 프로토콜은 LBRM (Log-Based Receiver-reliable Multicast) 등이있다[5]. 서버기반 오류제어방식에서 각 서버는 송신자의 모든 데이터를 기록하고있다. 서버는 성공적인 데이터수신에 대해 송신자에게 ACK 메시지를 보낸다. 수신자는 데이터손실에 대해 송신자가 아닌 서버에게 NAK 메시지를 보낸다. NAK 메시지를 받은 서버는 서버에 저장해놓은 데이터를 이용하여 해당 수신자에게 데이터를 재전송 한다. 이 방식은 여러 개의 서버를 두어 트래픽 부하를 분산시키는 장점이 있으나, 별도의 서버선정 및 관리, 그리고 추가적인 메모리할당 등의 오버헤드를 요구한다.

트리기반 local recovery 방식에서는 전체 수신자 그룹을 논리적인 트리형태로 구성하여, 각 sub-tree에 해당하는 local group 별로 오류제어를 담당하게 한다. 이로써 송신자의 처리부담을 골고루 분산시킬 수 있음은 물론, 많은 수신자에 대해서도 높은 확장성을 갖는다. 이러한 트리기반 오류제어기법을 사용하는 프로토콜에는 TMTP (Tree-based Multicast Transport Protocol)^[6] 및 RMTP (Reliable Multicast Transport Protocol)^[7] 등이 있다. 각 수신자(child)는 성공적인 데이터수신의 경우 root 혹은 parent에게 ACK 메시지를 보낸다. 이러한 측면에서 제어용트리를 ACK 트리라고도 부른다. 각 sub-tree의 루트는 자신이 관할하는 child 들의 데이터 수신 상태를 취합하여 상위루트에게 주기적으로 전달한다. 데이터손실의 경우 수신자는 자신의 parent에게 NAK를 유니캐스트로 알리거나 (RMTP의 경우) 혹은 sub-tree내에 멀티캐스트로 전달한다 (TMTP의 경우).

데이터 재전송의 경우, 수신된 NAK 메시지 수에 따라 유니캐스트 혹은 멀티캐스트방식으로 수행된다. 트리기반 오류제어기법은 매우 좋은 확장성을 갖는다^[1]. 하지만 네트워크계층의 트리와는 별도의 제어용 트리를 구성 및 관리해야 하는 오버헤드가 요구된다. 또한 단일세션이라 하더라도 송신자가 여러 명 있는 경우, 여러개의 트리가 구성되어야 한다. 한편, Lorax^[4] 프로토콜에서는 단일 멀티캐스트 세션에서 여러 송신자가 존재하는 many-to-many 서비스를 위해 공유형(shared) ACK 트리방식을 제안하였다.

모든 응용서비스에 적합한 오류제어기법을 개발하기란 매우 어려운 일이다. 실제 최근의 연구동향을 살펴보면, 특정응용서비스 혹은 특정 목적에 적합한 오류제어기법을 개발하고자 하는 노력을 찾아볼 수 있다. 이에 해당하는 예로써는 MFTP (Multicast File Transfer Protocol)^[5], STORM (Structure-Oriented Resilient Multicast)^[6], PGM (Pretty Good Multicast 및 Pragmatic General Multicast)^[7] 프로토콜, 그리고 ALF (Application Level Framing)^[8] 접근 방식 등이 있다.

III. FEC기반 오류제어 메카니즘 제안

1. 기존 FEC 기반 오류제어기법

최근 대규모 고속 데이터망의 출현과 멀티미디어 데이터 전송요구의 증대로 신뢰성 있는 데이터 전송에 관한 필요성이 급속히 증대되고 있으며 오류제어 방식은 이러한 환경에서 신뢰성을 부가해주는 방법이다.

오류제어 방식으로는 데이터 재전송 방식(AHQ : Automatic Repeat reQuest)과 전방 오류 제어 방식(FEC)등이 사용된다. ARQ 방식이 수신자 측에서 오류가 발생하였을 경우 송신자 측으로 재전송을 요구하는 오류 제어 방식인 반면, FEC 방식은 에러정정 부호를 이용하여 발견된 오류를 수신자 측에서 정정하도록 하는 방식이다. FEC 방식은 그러한 특성으로 인해 수신자와 송신자 사이의 전송시간이 크거나, 음성통신과 같은 잡음이 어느 정도까지 허용되고, 데이터의 실시간 성이 요구되는 환경에서 사용하는 것이 유리하다^[9]. 이장에서는 FEC의 기본개념과 FEC를 응용한 관련 기법들에 대해 서술한다.

1) FEC 기법의 개요

원래의 데이터를 일정한 비트 수 k (≥ 1)로 구성되는 데이터의 블록으로 나누어서 부호화를 수행하면 각 블록은 2^k 개의 블록 x (≥ 1)가 된다. 부호화기는 블록 x 에 0 패리티검사 비트를 첨가하여 길이가 n 인 블록 y 로 생성해 내게 되며, 이때의 생성된 결과 y 는 전송 중에 발생하는 에러를 검출할 수 있게 한다. 이와 같은 방식은 (n, k) 로 표시하며 패리티 검사 비트는 $n-k$ 개가 되고, $n > k$ 가 성립한다.^[10]

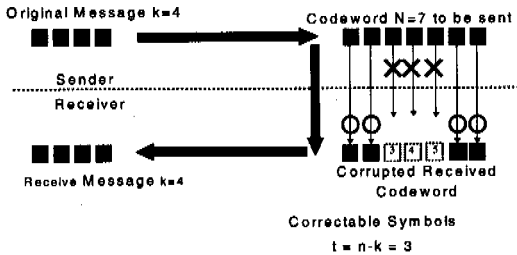


그림 1. FEC 기법

수신 측에서는 복호화 과정을 통해 원래의 데이터를 복구해 낼 수 있으며, 이 과정에서 오류가 발생하였는지의 여부도 알아낼 수가 있다^{[11][12]}.

이러한 과정은 부호화 과정을 통해 오류복구 정보를 포함하게 함으로써 수신 측에서 오류가 발생하였을 경우 자체적으로 복구할 수 있도록 함으로써, 오류가 발생하였을 경우 일반적으로 재전송을 요구하는 ARQ방식에 비해 피드백요구와 재전송을 낮출 수 있게 해 준다. 수신 측에서는 오류정정을 위한 여분의 데이터를 부호화를 통하여 추가하게 되고 생성된 데이터를 수신 측에 전달하게 되며, 수신 측에서는 복호화를 통하여 원래의 데이터를 복구해 내게 된다. 복호화 과정은 원본 데이터를 얻어내는 과정뿐만 아니라 오류가 일어났는지를 파악해 낼 수 있으며 오류가 존재하는 경우라도 정정이 가능한 범위 내에서는 원본 데이터를 얻어낼 수가 있다. 오류가 발생하였다더라도 복호화를 통하여 원래의 데이터를 복구해 내도록 하는 FEC의 부호화 정도를 조정해 줌으로써, 전송된 데이터가 손상되었다라도 수신자 측에서 오류복구를 시도하고 복호화도 복구해 낼 수 없는 경우에만 상위 수신자에게 재전송을 요구하는 방법을 이용하여 재전송의 수를 감소시킴으로써 피드백 폭주 문제를 완화시킬 수 있으며 FEC 기법의 적용 범위를 설정하기 위해 계층구조를 이용하여 FEC 기법만의 단점을 극복하도록 한다.

2) FEC 기법의 유형

기존 FEC 기법에는 ARQ 전송방식 내에 FEC 전송방식을 포함시켜 구성함으로써 ARQ 방식과 FEC 방식의 장점만을 취한 형태의 Hybrid ARQ, Reliable Multicast 환경에서 많은 수의 동종 수신자들에게 파일을 전송할 수 있는 프로토콜로서 내부적으로는 소프트웨어 적인 FEC기법을 사용하는 RMDP(Reliable Multicast data Distribution Protocol), Reliable Multicast가 그 수신자 영역이 넓어짐으로 인한 신뢰성 저하 문제를 해결하기 위한 방법인 SHARQFEC(Scoped Hybrid Automatic Repeat reQuest with Forward Error Correction), 데이터 손실에 대한 문제들은 재전송의 한계문제와 원하는 수신자에게 재전송이 제대로 이루어질 수 있는가 하는 문제들을 발생시키며 이러한 문제들을 해결하기 위해서 두 가지 일반적인 복구 기술을 동시에 적용한 APES(Active Parity Encoding Services protocols) 등이 있다.

Hybrid ARQ는 송신 측에서는 데이터를 전송할 때 여분의 데이터를 포함하여 부호화 한 결과를 전송하고 수신측 복호화 과정에서 오류가 발생하였음을 감지하였을 경우 오류 정정을 시도하고 여분의 데이터를 이용한 복호화라도 정정이 불가능한 경우에만 재전송을 요구하도록 함으로써 신뢰성을 높일 수 있다. 그와 같이 ARQ와 FEC를 조합한 Hybrid 방식은 ARQ만을 이용했을 때 보다 더 높은 신뢰도를 얻어낼 수 있다.

RMDP는 수신자가 서버가 요청하는 파일에 접근해서 수신자가 서비스 받기를 원하는 요구 사항을 기술하면 송신자가 서비스 가능한 명세를 담아서 응답하고 난 후에 실제 포용 가능한 것 보다 조금 더 여유 있게 전송한다. 수신자 측에서 전송과정에 문제가 없다고 판단되는 경우에는 더 높은 전송률을 요구한다. 하지만, 이러한 RMDP 기법은 수신자 쪽에서는 수신자의 정보를 저장하고 있지 않기 때문에 수신자 측의 요구가 손실되는 경우에는 그 수신자는 요구대상에서 제외된다는 점과 모든 수신자들에게 오류 정정을 위한 추가적인 정보가 필요한 것은 아니라는 단점을 가지고 있다.

SHARQFEC는 지역내의 계층적인 구조를 이용한다. 수신자들의 거리를 간접적으로 결정하기 위해 새로운 방법을 사용하여 확장성에 따르는 문제를 줄이고 FEC는 재전송을 줄이기 위한 방법으로 이용한다. 수신자는 오류가 발생한 경우 무조건 송신자에게 재전송을 요구하는 것이 아니라 자신이 속

한 지역 내에서 해결을 하려고 시도하게 된다. 현재 자신의 지역에서 오류복구가 불가능한 경우에는 상위 지역으로 정정 요구를 전달한다. 오류 복구 과정은 두 가지 방법으로 수행되는데 첫 번째는 ARQ와 비슷한 방법으로 NACK를 수신하면 추가적인 FEC 패킷을 보내주는 방법이고, 두 번째는 NACK가 도착하지 않아도 무조건 잉여 패킷을 보내주는 방법이다. 보내주게 되는 패킷의 수는 그 수신자의 패킷 손실률에 따라 결정이 된다.

APES에서 적용된 기술은 (Active) Repair Server와 Parity 부호화(FEC)이며 이중 Repair Server는 오류가 일어나는 상황에 대해 적절히 대처하는 역할을 담당하고 Parity 부호화는 오류발생 빈도를 줄이는 역할을 수행한다. APES적용 방법과 Repair Server의 역할에 따라 SDBR(Store-Data-Build-Repairs), BRSR(Build-Repairs-Store-Repairs), GRSR(Get-Repairs-Store-Repairs) 과 같은 방법들이 있다.

2. 고정된 오류제어기법의 문제점

네트워크 상황에서 일부 노드에 문제가 발생하였을 경우나 하위 수신자들이 가지고 있는 문제가 감당하기 힘들어 지는 경우가 발생할 수 있으며, 그로 인해 중간 노드에서 문제가 생기게 되면 모든 하위 수신자들도 같은 영향을 받게 되며 상위 계층까지로 문제가 연속적으로 발생되어 올라가게 될 수도 있다. 또한 문제는 연속적으로나 규칙적으로 발생하는 것이 아니므로 어떤 곳에서도 발생할 수 있으며 문제가 발생하는 계층이 상위 계층일수록 그 영향의 범위가 커지게 된다. 이 경우 문제 해결을 송신자에게서만 기대하게 되면 송신자에게 피드백 폭주 문제를 비롯한 여러 가지 문제가 발생하며 문제점 해결을 위한 지연시간도 생기게 된다. 또한 송신자가 오류 정정 기법을 변경하는 경우 모든 수신자들에게 그 사실이 통보되어야 하며 모든 수신자들이 변경된 기법에 적용되게 됨으로써 오버헤드가 발생하거나 연속적으로 문제가 발생하는 경우에 그 대처방법이 적절치 않게 될 수가 있다. 일반적으로 네트워크 상황이 좋아서 오류발생빈도가 적은 경우에는 ARQ가, 오류발생비율이 높은 경우에는 자체로 오류를 정정할 수 있는 코드가 높은 효율을 나타내며^[13], 수신자 수가 적은 경우에는 ARQ가, 수신자 수가 많은 경우에는 FEC가 더 높은 성능을 보인다^[14]. 하지만, 네트워크 상황은 시간적으로나 지역적으로 그 성능이 균등하지 않고 유동적이기 때문에 일률적인 한 방법만을 적용하기에는 곤란하다.

3. Adaptive FEC 오류제어기법

본 논문에서는 계층구조를 사용하며 계층구조는 FEC를 차등적으로 전송하기 위한 나름대로의 영역을 제공하게 된다. 계층구조의 적용으로 인하여 수신자들을 상위 계층과 하위 계층을 구분할 수 있게 되며 문제가 발생하는 경우 중간계층에서 해결을 위한 동작을 수행하게 된다.

네트워크를 지역내로 들어오기 이전까지의 상위 계층과 지역내의 하위 계층으로 나누어 각 계층의 상황에 따라 차등적인 방법으로 전송하도록 하는 것이 상황에 맞게 좀 더 효율적으로 대처할 수 있게 한다.

본 논문에서 제안하는 Adaptive FEC(이하 적응 FEC)방법은 문제 발생을 접수하는 중간계층에서 하위 계층구조 단위로 오류가 적게 일어나는 경우 낮은 FEC기법을 적용하고, 상대적으로 오류가 빈번하게 일어나는 경우에는 높은 FEC기법을 독립적으로 적용하여 전송함으로써 전체 전송을 모두 FEC로 적용하거나 하지 않는 방법보다는 재전송의 수를 적절히 조정하면서도 원하는 만큼의 전송률을 보장할 수 있게 한다.

계층구조를 적용함으로써 각각의 계층에는 중간 계층이 생기게 되며 하위 계층을 관리하는 중간 계층은 어떠한 중간 수신자라도 될 수가 있도록 하여

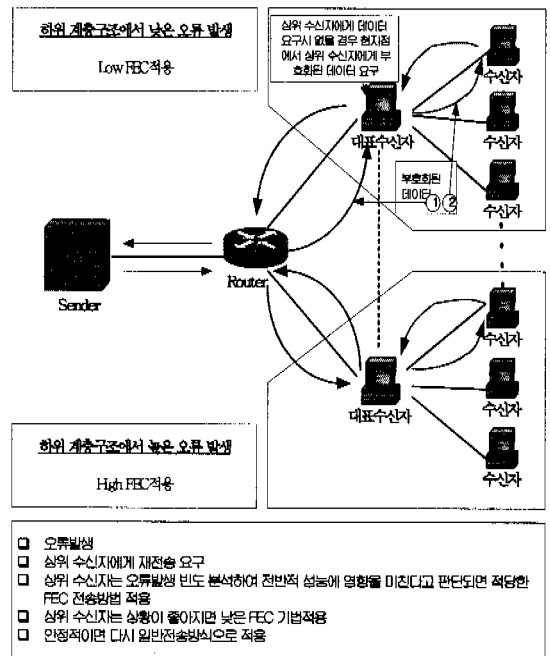


그림 2. Adaptive FEC 방법

오류 발생률에 따른 적용범위가 유동적으로 적용되도록 한다. 상위수신자와 하위수신자들에게는 각각의 쌍으로서 독립적인 개념으로 작용하게 된다. 하위수신자들은 상위 수신자에게서 전송 받은 일반적인 데이터가 오류가 발생하였을 경우 상위수신자에게 재전송을 요구하게 되고 상위계층은 이런 하위수신자들의 오류발생빈도를 분석하여 그것이 전반적 성능에 영향을 미친다고 판단될 경우 FEC기법을 변환하여 전송하게 된다. 변경된 FEC기법이 적용된 후부터는 계속 변경된 FEC기법이 적용되기 때문에 bulk data가 갑자기 폭주한 경우나 오류가 지속적으로 발생하는 경우에 적용성 있게 대처하도록 한다. 상위 수신자는 전송을 하면서 그 상황이 좋아졌다고 판단될 경우에는 상대적으로 낮은 FEC 방식을 적용하여 수행하며 안정적인 상황이라고 판단되면 일반적인 전송방식으로 전환한다

1) 수신자의 복호화 방법

기법이 변환되는 과정에서 중요하게 고려해 주어야 할 문제는 기법이 변한다는 것을 하위 수신자에게 고지를 해야 한다는 것이다. 상위 계층의 일반적인 기법 변경이나 변경 메시지를 제대로 수신하지 못하여 하위 수신자가 기법의 변화를 알 수 없게 되는 경우, 부호화 방법이 바뀐 데에 대해 그 이전의 복호화 방법이 계속 적용되므로 전송되는 데이터가 연속적으로 오류를 일으킨 것으로 잘못 인식하게 된다. 이런 문제를 해결하기 위해서 일반적으로 안전한 연결설정을 통해 메시지를 전달하거나 모든 하위 수신자들이 해당 메시지를 제대로 수용했는지 피드백 정보를 수집해야 하며 이러한 방법은 모든 수신자들의 피드백 정보가 모아질 때까지 기다려야 한다는 문제점이 있다. 또한 전송되는 시점을 맞추기 위해 동기화를 시행하거나 일정한 간격을 두어 부호화와 복호화 시점을 맞추어 주어야 한다는 단점이 생기게 된다.

본 논문에서 제안한 적응 FEC 방법은 다른 기법을 적용하게 되는 때 경우마다 메시지의 길이가 달라지게 되고 그러한 메시지를 받는 수신자 측에서는 전달되는 메시지의 길이를 보고 복호화 방법을 선택하도록 한다. 수신되는 메시지의 길이에 따라 복호화를 달리하는 기법은 추가적인 수정메시지의 교환을 통해 이루어지는 방법보다는 메시지처리를 위한 추가적인 고려가 복잡하고 까다롭지 않게 진행된다. 추가적인 메시지를 이용하지 않기 때문에 상위 수신자가 굳이 그 방법이 변경됨을 고지하지

않아도 되므로 메시지가 제대로 전달되지 않는 경우를 고려하지 않아도 되며, 모든 수신자들이 기법이 변경되었다는 것을 제대로 알았는지 확인하지 않아도 되기 때문에 수신자들로부터의 피드백을 받음으로 인해 생기는 지연시간이 생기지 않는다. 추가적인 지연이 생기지 않는다는 것은 문제가 생겼을 경우에 좀 더 빠르게 대처가 가능하다는 의미를 포함하고 있다. 만약 데이터 전송과정에서 일어나는 메시지의 손실로 인해 데이터길이의 변화가 생기는 경우에도 복호화 과정에서 오류로 인식되므로 오류가 발생했음을 자동으로 감지/인식해 낼 수 있다.

2) 오류 발생률에 따른 적응 FEC 적용

중간계층에서는 상위로부터 데이터를 전달받아 하위로 전송해주는 기본적인 역할을 하면서 하위수신자들로부터 오는 재전송요구 메시지를 상위 수신자에게 전달하는 역할도 한다. 이때 상위 수신자에게 재전송요구 메시지를 전달하는 과정에서 현재의 전송기법이 적절치 않다고 판단되는 경우에는 자체에서 기법의 변경을 선택하게 된다. 적응 FEC 기법은 중간 계층에서 하위 수신자들의 피드백 정보를 수신한 후 평가하여 적절하게 FEC 정도를 변경하는 기법으로서 수신자들의 상태에 맞는 적절한 방법을 선택하도록 하는 것이 어느 중간 노드에서나 독립적으로 일어날 수 있다. 상황에 맞게 적절하게 FEC 정도를 변경하여 줌으로써 필요이상의 오버헤드로 작용될 수 있는 FEC 기법의 약점을 보완하도록 한다.

가) 오류 발생률이 High Rate인 경우

오류 발생률이 높아져서 재전송 요구가 많아지게 되면 대표 수신자에서 적절하게 다음 그림과 같은 미리 정의된 기법중에서 부호화 정도 즉 Repair Packets율이 높은 것을 적용하게된다.

미리 정의된 방법을 사용함으로써 수신자측에 별도의 기법 변경 메시지를 통보하지 않고 수신자측과 송신자측에서 가지고 있는 정의화된 규칙에서 부호화 정도의 크기에 따라 기법을 변경하여 적용할 수 있으므로 기존의 고정적인 방법보다는 훨씬 적용이 용이하다.

나) 오류 발생률이 Low Rate인 경우

오류 발생률이 높아지는 경우와 달리 낮아지는 경우 중에 현재 선택되어진 FEC 기법보다 더 낮은 단계의 FEC 기법만으로도 오류가 발생하는 상황에 충분히 대처가 가능한 경우에는 현재의 FEC 기법

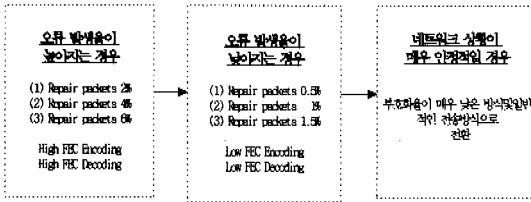


그림 3. 전송모드 방식의 변경 및 조정

중에 오류 정정 코드로 삽입된 부분에서 더 낮은 단계의 기법의 추가적인 코드를 뺀 것만큼이 부담으로 작용하게 된다. 이러한 부담은 다수의 수신자를 상대로 하는 멀티캐스트의 특성상 수신자의 수가 많아지면 많아질수록 트래픽 양의 증가로 이어지게 된다. 오류가 상대적으로 낮게 발생하는 경우 FEC 기법을 변경하여 효율을 피하는 방법을 취하여야 하며, 현재의 상태가 더 낮은 단계의 FEC 기법으로도 적용이 충분치 가능하다고 판단될 경우 기법을 변경하여 전송하게 된다. 기법을 변경하게 될 때에는 오류 발생률이 높아지는 상황에서의 대처법과 마찬가지로 하위수신자들에게 특별한 의사메시지를 전달하지 않고 다른 기법을 적용한 이전과는 다른 길이의 부호화 된 데이터를 생성하여 전달하게 됨으로써 하위 수신자는 상위 수신자에게 받은 데이터의 길이를 통해 복호화 방법을 선택하게 된다. 낮은 기법으로 변환하는 경우 높은 오류 정정 데이터를 추가적으로 삽입하게 하는 기법으로의 변환보다는 훨씬 더 신중하게 결정이 되어야 한다. 잘못된 판단으로 인해 적절치 않은 상황에서 낮은 기법으로 변환한 경우 일시적으로 오류가 폭증해서 다른 중간노드들에게도 잘못된 판단을 하도록 영향을 미칠 수 있기 때문이다. 잘못된 판단이 적용이 되어 재전송요구가 많아지더라도 어느 정도의 시간이 지나면 다시 적절한 상태로 복귀될 수 있지만 멀티미디어 데이터처럼 실시간 전송이 강조되는 경우 지연시간이 생김으로 인해서 서비스 품질에 신뢰성을 저하시킬 수 있다. 따라서 기법의 변경은 오류 발생률이 낮아지는 경우 서서히 강도를 낮추어서 적용할 필요성이 있다.

다) 데이터 전송모드의 조정 및 변경

데이터를 전송하는 과정에는 단순히 상위 수신자로부터 전달받은 데이터를 하위 수신자에게 재전송해주는 과정뿐만이 아니라, 현재 수신된 데이터에 적용된 부호화 방법이 적당하지 않다고 판단되는 경우 새롭게 부호화 방법을 적용하는 과정도 포함

이 되어 있다. 이러한 과정은 하위 수신자들로부터 재전송 메시지가 도착하였을 경우 기법을 변경할지를 판단하여 설정한 변수를 이용하여 현재 수신된 기법과 변수의 값에 차이가 있을 경우 새로운 부호화 방법을 수행하여 얻어낸 결과를 전송하게 된다. 기법을 변경하는 경우 임의적으로 기법을 선택하는 것이 아니고 그림 3 과 같이 몇 가지 방법을 정해 놓은 범위 안에서 선택을 하게 된다. High FEC나 Low FEC로 변경을 선택하는 경우 그것이 선택할 수 있는 허용치를 벗어나지 않는지를 판단하여 벗어나지 않는 경우에만 새로운 방법을 선택하도록 한다. High FEC을 이용하여 전송하는 상황에도 자주 High FEC 모드로의 변경을 수행하였다가 실패하지 않기 위해서 선택할 수 있는 최상위 모드와 최하위 모드를 적절하게 선택하는 것이 중요하다.

3) 대표수신자의 선정

지역 네트워크 수신자들 가운데서 하나의 대표수신자를 선정해야 하는 과정에서는 동적인 대표수신자 방식을 사용한다. 여기에는 새로운 지역네트워크가 발생하는 경우와 기존 대표수신자이 종료로 기존 지역네트워크에서 새로운 대표수신자로 선정해야 하는 경우가 있다. 새로운 지역 네트워크 수신자가 발생할 경우는 일정시간 지역 대표수신자의 존재 여부를 확인하여 응답이 있으면 현상태를 유지하고 없다면 최초 수신자가 대표수신자가 된다.

대표수신자 종료로 인한 새로운 대표수신자 선정은 우선 지역 대표수신자 종료 메시지가 있는지를 일정시간 조사하여 존재하면 지역 네트워크 안에서 낮은 IP를 가지고 있는 것이 새로운 대표 수신자가 된다.

IV. 시뮬레이션

본 논문에서 제안한 적응 FEC 메커니즘을 알아보기 위해서 기존의 재전송 기법인 ARQ기법만을 이용한 방법과 정적인 FEC기법을 사용하는 경우를 그리고 적응 FEC기법을 사용하여 수신자들의 변화에 의한 각 수신자들의 재전송 수행횟수와 성능, 전송 패킷 수 변화에 의한 각 수신들의 재전송 수행 횟수와 성능, 오류율 변화에 의한 각 수신들의 재전송 수행횟수와 성능을 실험한다.

시뮬레이션은 Linux 환경의 워크스테이션과 일반 PC Windows2000환경에서 C언어와 C++언어로 작성되었으며 시뮬레이션을 위한 개체는 송신자, 중간 수신자와 수신자로 구성하여 가상의 멀티캐스트 트

리를 만들어 수행하였다. 시뮬레이션에 사용된 고정 FEC 코드는 Luigi Rizzo의 98년 6월 개정된 코드를 사용하였으며 Adaptive FEC코드는 Luigi Rizzo의 코드에 본 논문에서 제안하고 있는 알고리즘을 적용하여 구성하였다^{[15][16]}.

그림 4는 Adaptive FEC 알고리즘의 순서도이고 그림 5는 이 실험에 사용된 Topology이며 15개의 노드로 구성이 되어있다. 실험의 간결성을 고려하여 더 많은 노드를 추가하여 실험 할 수도 있지만 정확한 실험 값을 위하여 Level 4까지의 서브트리를 구성하는 방안을 사용하였으며, 실험항목 3에서의 수신자수는 20과 100이 사용하였는데 이것은 15개까지의 실험에 의한 실험 예측값을 근거로 하여 시뮬레이션 된 값이다.

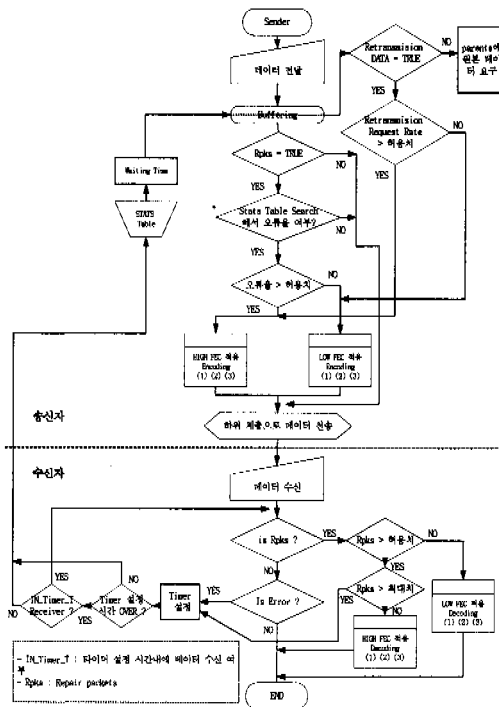


그림 4. Adaptive FEC 알고리즘 순서도

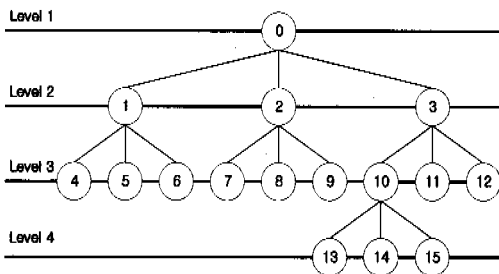


그림 5. 시뮬레이션 Topology(평면도)

이 실험에서 측정하고자 하는 것은 변수로 주어진 수신자수, 패킷수, 오류율의 변화에 따른 ARQ, FEC, Adaptive FEC 방식들 간의 재전송 횟수를 통한 비교 우위 관계를 그래프로 표현하였다.

Topology를 통하여 측정된 사항은 다음과 같은 사항들이다.

표 1. 실험 측정 항목

	실험 1	실험 2	실험 3
수신자수	변화(3, ..., 15)	고정(15)	20, 100
패킷수	고정(20)	변화	고정(20)
오류율	2%, ..., 6%		변화(2%, ..., 8%)

실험항목 1에서는 수신자수를 3~15개로 3, 12, 15개 단위씩 각각 설정한 상태에서 20개의 데이터를 보내는 것으로, 오류율이 2%, 4%, 6% 인 상황에서 수행하였으며 2%인 경우를 보면(본지에서는 지면 관계상 2%만을 표시함) ARQ 방법은 수신자 수 증가에 따른 오류에 대한 재전송 요구 횟수가 다른 FEC 방법보다 크며 또한 FEC 방법은 제안한 적용 FEC 방법보다 재전송요구 횟수가 다 많음을 알 수 있다.

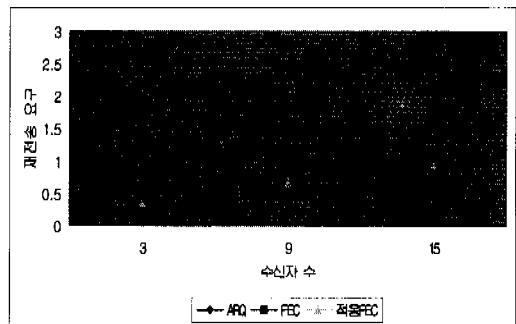


그림 6. 실험항목 1 (오류율 2%)

실험항목 2에서는 수신자수 15개에 대한 오류율이 2%, 4%, 6%인 상황에서 데이터를 20~100개씩 20개 단위로 변화시켜서 전송하는 것으로 수행되었다.

이 실험에서도 ARQ 방식이 패킷의 변화에 대한 재전송요구는 ARQ, FEC, 적용 FEC 순으로 나타남으로서 적용 FEC방식이 다른 두 방식보다 패킷의 증가에 따른 재전송 요구 횟수가 적게 나타나

로 그림 5의 Topology에서 1, 2, 3 노드에 대한 폭주 및 피드백 요구에 의한 부하가 보다 더 적게 받을 수 있는 환경을 제공할 수 있다.

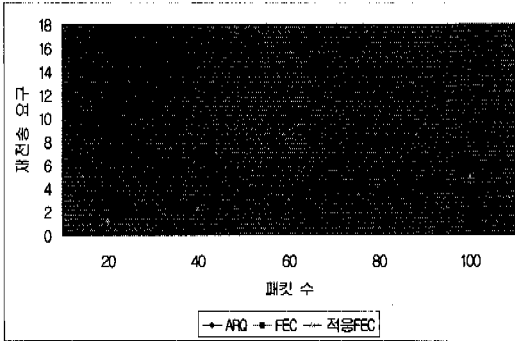


그림 7. 실험항목 2 (오류율 2%)

실험항목 3은 각각의 경우 매 2%단위의 오류 발생률을 갖도록 설정하고 데이터 패킷 20개에 대한 수신자수를 20개와 100개로 하여 수행하였으며 100개에 대한 데이터값(본지면에서 20개에 대한 데이터 값은 지면 관계상 생략)을 표시하였다. 실험 그래프는 ARQ 방법이 기본 데이터의 크기가 5배로 증가하였을 때 상당한 재전송 요구를 보이고 있으며 FEC와 적응 FEC를 비교 할 때에는 점점 기술기 차이가 패킷 수 증가에 따라 차이가 발생함을 알 수가 있다. 이 실험으로 BULK data 에 대한 적응 FEC 방법이 다른 방법보다 재전송 요구횟수가 적게 나타남을 알 수가 있다.

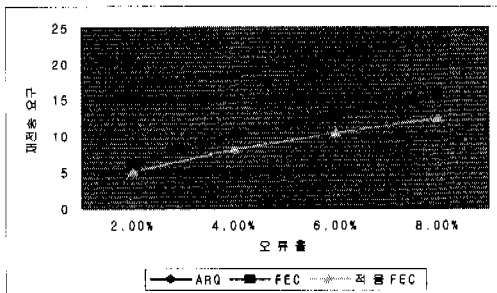


그림 8. 실험항목 3 (수신자수 : 100)

V. 결론

통신망의 발달에 따른 데이터의 증가는 어떻게 데이터를 효율적으로 전달하고자 하는 곳까지 데이터의 성질을 변화되지 않고 오류없이 전달할 것이

냐는 문제가 제기되고 있고 또한 이러한 신뢰성 보장에 관한 연구가 진행되고 있다. 이러한 신뢰성에 대한 문제는 UDP 기반의 멀티캐스트에서는 더욱 필요하다 할 수 있다. 하지만 많은 수신자가 존재하는 멀티캐스트에서 Classical Method로 접근하였을 때 피드백 폭주 및 대용량의 데이터를 전송할 때 발생하는 지연 등의 문제가 야기 될 수 있다.

그와 같은 문제를 해결하기 위해 수신자 측에서 오류가 발생하였음이 감지되었을 경우 그 자체적으로 에러를 복구할 수 있도록 하는 FEC를 적용하게 된다. 하지만, FEC 기법만 적용하였을 경우 네트워크 환경에서 오류가 적게 발생된다면 그만큼의 여분의 데이터가 부담으로 작용하게 된다는 문제점이 생기게 된다.

본 논문에서는 멀티미디어 데이터가 다양한 수신자에게 전송이 되는 멀티캐스트 환경에서의 피드백 발생의 빈도와 재전송 횟수를 줄이기 위한 방법으로 FEC를 사용함으로써 오류와 재전송을 효율적으로 줄일 수는 있지만, FEC 부호화과정과 부호화 된 데이터가 네트워크에 부담으로 작용될 수도 있는 상황을 해결하기 위해 네트워크를 계층적인 구조로 구성하고 FEC를 적절히 조절하여 지역적으로 독립적인 방법을 적용하는 방법을 제안하였으며, 고정적인 FEC를 이용하는 방법보다 적절한 수준으로 재전송 수를 감소시킬 수 있음을 보여 주었다. 단, 모델을 어떻게 구성하는가와 어떤 판단 기준을 가지고 기법을 변경할 것인지를 판단하는 파라미터를 설정해 주는 것이 성능에 영향을 미칠 수 있기 때문에 적절한 모델 구성과 효율적인 파라미터를 설정해 주는 것이 중요하다.

앞으로의 연구 과제로는 다양한 모델 및 설정상황의 변경에 대한 문제의 검증과 다양한 환경에서의 실험이 필요하다.

참고 문헌

- [1] K. Obraczka, "Multicast Transport Protocols: A Survey and Taxonomy", IEEE Communications Magazine, pp. 94-102, January, 1998.
- [2] Jean-Chrysostome Bolot, Thierry Turletti and Ian Wakeman, "Scalable Feedback Control for Multicast, Video Distribution in the Internet", ACM SIGCOMM '94, pp58-67, August 1994.
- [3] B. Whetten, L. Vicisano, R. Kermode, M. Handley, S. Floyd, "Reliable Multicast Trans-

