논문 13-38A-07-08

한국통신학회논문지 '13-07 Vol.38A No.07
http://dx.doi.org/10.7840/kics.2013.38A.7.612

다중 홉 MANET에서의 파일 전송 응용을 위한 코딩 패킷 그룹 기반 ARQ 기법

김 영 실◆, 강 경 란*, 조 영 종*

A Coding Packet Group-Based ARQ Scheme for File Delivery Services in a Multi-Hop Wireless Network

Youngsil Kim◆, Kyungran Kang*, Young-Jong Cho*

요 약

본 논문에서는 이동성이 없는 다중 홉 무선 네트워크 환경에서 파일 전송을 위한 신뢰성 있는 코딩 패킷 그룹 기반 네트워크 코딩 (Group-based Reliable Network Coding, rNC) 기법을 제안한다. rNC는 소스 노드와 목적지 노드 간에 멀티-홉 네트워크 코딩 노드들을 고려하여 설계되었다. 각 네트워크 코딩 노드는 큐 관리 기법으로 폴링 시스템을 적용하여 일정 시간 동안 수집된 패킷들을 하나의 코딩 패킷 그룹으로 정의하고 이들을 랜덤 선형 네트워크 코딩 기법을 사용하여 전송한다. 네트워크 코딩 노드들 간에는 코딩 패킷 그룹 단위의 신뢰성 있는 전송을 추구한다. 소스 노드는 자신의 다음 네트워크 코딩 노드로부터 자신이 정의한 코딩 패킷 그룹들에 대한 수신 완료를 수신하면 데이터 전송을 완료할 수 있다. ns-2를 활용하여 시뮬레이션을 통해 제안하는 기법의 성능을 평가하였다. 잘 알려진 CodeCast과 rNC의 성능을 비교 분석하였다. 시뮬레이션 결과는 네트워크를 구성하는 링크의 에러율이 높아질수록 rNC가 CodeCast 보다 높은 패킷 전송률을 보였다. 또한, 소스 노드의 파일 크기가 증가함에 따라 rNC는 CodeCast 보다 더 낮은 네트워크 코딩 지연 시간 증가를 보였고 적은 네트워크 부하를 발생시켰다.

Key Words : network coding, coding group-based ARQ, file delivery, polling systems, multi-hop wireless network

ABSTRACT

In this paper, we propose a coding packet group-based ARQ scheme (rNC) for file delivery in wireless networks. rNC assumes multiple network coding points between the source and the destination. Each network coding point gathers and codes a group of packets according to the queue polling system. A queue polling system makes a few or several packets available for coding in a queue while polling the other queues in the system. Thus, we assuem a queue polling system at each network coding point. We call this group of packets as coded packet group. Each coding point acknowledges the reception of every code packet group to its previous coding point for reliable delivery. Thus, the intermediate coding points including the source can release its buffer before the packet is delivered to the destination. To guarantee the ultimate file delivery to the destination, the destination sends acknowledgement to the sender. We evaluate our proposed scheme using ns-2 and compare the performance with CodeCast. The results show that rNC works better than CodeCast in terms of packet delivery ratio and control overhead in unreliable wireless networks.

※ 본 연구는 미래창조과학부 및 정보통신산업진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음 (NIPA-2013-1415128749)
◆ 주저자 : 아주대학교 컴퓨터통신연구실, yskim@ajou.ac.kr, 정회원
* 아주대학교 컴퓨터통신연구실, korykang@ajou.ac.kr, 정회원, yjcho@ajou.ac.kr, 종신회원
논문번호 : KICS2013-03-142, 접수일자 : 2013년 3월 28일, 최종논문접수일자 : 2013년 6월 19일

Ⅰ. 개 요

무선 네트워크에서 데이터 전달의 신뢰성 보장은 중요한 이슈이며 지금까지 많은 연구가 이뤄지고 있다[1,2]. 무선 네트워크의 신뢰성을 떨어뜨리는 주요 원인은 fading, interference 그리고 collision 등이 있으며 이들로 인해 무선 멀티캐스트 네트워크에서 소스는 수신자들에게 신뢰성 보장을 하기 어렵다. ARQ(automatic repeat request)는 데이터 전달의 신뢰성을 보장하기 위한 기법으로 데이터 전달이 실패했을 경우 재전송을 함으로써 데이터 전달의 신뢰성을 높인다[3,4]. 신뢰성이 낮은 무선 네트워크에서 ARQ 기법은 잦은 재전송 발생으로 'ACK storm'을 야기하고 많은 양의 불필요한 전송 기회를 낭비하게 된다. ACK 부하를 줄이기 위한 방법으로 ARQ와 FEC를 결합한 기법이 있다[5,6]. FEC 기법은 무선 네트워크의 간접듣기(overhearing) 특징을 고려하지 않은 것으로 멀티-홉 네트워크에서 중간 노드들 간의 중복 전송을 야기하여 대역폭을 낭비하게 된다.

네트워크 코딩 기법은 무선 네트워크의 효율적 대역폭 사용을 가능하게 하는 것으로 잘 알려 져 있다[7]. 기존의 네트워크 노드가 단순히 패킷을 저장하고 전달하는 형태의 기능을 했던 것과는 달리 네트워크 코딩은 네트워크 코드들이 패킷들은 인코딩하여 전달한다. 신뢰성, 지연시간, 에너지 등의 관점에서의 네트워크 코딩의 이점은 이미 여러 연구에서 증명되었다[8,9,10,19].

무선 네트워크에 네트워크 코딩을 도입하여 데이터 전달의 신뢰성을 보장하기 위한 기법들이 다수 제안되어 있다[11-14]. R-Code[11]는 무선 매쉬 네트워크 환경에서 신뢰성 있는 패킷 전달을 목적으로 하며, AdapCode[12]는 네트워크 오버헤드와 메모리 사이즈를 고려한 무선 센서 네트워크 환경에서 낮은 트래픽 발생, 짧은 지연시간, 그리고 효과적인 부하 균형을 목적으로 하는 기법이다. GreedyCode[13]는 무선 메쉬 네트워크에서 신뢰성 있는 데이터 전달을 목적으로 하고 있다. CodeCast[14]는 다중 홉 MANET에서 신뢰성 있는 데이터 전달을 위해 제안되었다. 특히, CodeCast는 코딩된 패킷들을 고정된 크기의 블록들로 나누어 전송한다는 점에서 본 논문에서 제안하는 기법과 유사하다.

본 논문에서 제안하는 기법 *r*NC는 다중 홉 MANET에서 파일을 멀티캐스트하는 경우에 신뢰성 있게 전달하는 것을 보장하며 불필요한 트래픽 부하를 최소화하는 것을 목표로 하고 있다. *r*NC는 네트워크 코딩 시에 동시에 인코딩되어 전송되는 패킷들의 그룹을 '코딩 패킷 그룹(coded packet group, 이하 CPG)'으로 정의하고, CPG의 송신자와 수신자 간의 ARQ 기법을 도입하여 패킷 전달율을 높인다. 본 논문에서는 소스 노드와 최종 목적지 노드 사이에 한 개 이상의 네트워크 코딩 노드가 존재한다고 가정한다. 전달 노드는 네트워크 코딩을 수행하지 않고 단순 전달 기능만을 수행한다. 소스 노드와 네트워크 코딩 노드가 네트워크 코딩을 수행하며 CPG 송신자와 수신자의 기능을 담당한다. 소스 노드는 일정한 크기의 CPG를 구성하여 패킷들을 조합한다. 네트워크 코딩 노드는 주어진 폴링 시스템 서비스 방식에 따라 소스 노드와는 다른 크기의 CPG 단위로 패킷들을 조합한다. CPG 송신자는 CPG 수신자로부터 각 CPG에 대한 ACK이 도착하기까지 랜덤 선형 네트워크 코딩 기법을 적용하여 코딩 패킷들을 전송한다. 소스 노드는 자신과 인접한 네트워크 코딩 노드로부터 ACK을 수신하면, 다음 CPG에 속한 패킷들에 대해 패킷 전송을 시작한다. 즉, 최종 목적지 노드로부터의 ACK을 수신하지 않더라도 새 패킷들에 대해 전송을 시작할 수 있으므로, 소스 노드가 최종 목적지 노드로부터 ACK을 수신하는 것에 비해 최종 전체 파일 전송하기까지의 소요 시간을 크게 단축할 수 있다.

본 논문에서 제시하는 기법과 유사한 선행 연구인 CodeCast와 성능을 시뮬레이션을 통해 비교 분석하였다. 시뮬레이션 결과, 제안하는 기법인 *r*NC가 CodeCast에 비해 네트워크 내의 링크들의 에러율에 대한 데이터 전송 성공률이 높았다. 또한, 소스 노드의 파일 크기가 증가 함에 따라 *r*NC는 CodeCast 보다 더 낮은 네트워크 코딩 지연 시간 증가를 보였다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구들을 살펴보고, 3장에서는 CPG 송수신 모델과 코딩 패킷 그룹을 전달 알고리즘 등이 설명되어 있다. 4장에서는 *r*NC 적용 네트워크의 성능 분석이 제시되어 있으며, 5장에서는 *r*NC과 CodeCast의 성능에 대한 비교분석 결과가 기술되어 있다. 마지막으로 결론에는 본 연구의 성과와 향후 연구 방향에 대한 내용이 언급되어 있다.

Ⅱ. 관련 연구

지금까지 무선 네트워크의 신뢰성 있는 멀티캐스

트는 피드백 과부하를 해결하기 위한 많은 연구가 이뤄지고 있다. 멀티캐스트의 재전송 과부하는 송수신 노드들 간의 ACK/NACK 전송과 송신 노드의 손실 패킷 재전송에 의해서 발생한다. ARQ 기법은 100% PDR을 보장하지만 ACK storm을 발생시켜 많은 양의 불필요한 네트워크 부하를 발생시킨다. 네트워크 코딩은 재전송으로 인한 네트워크 부하를 줄기기 위한 방안으로 제안되고 있다. 무선 네트워크의 신뢰성 있는 데이터 전송과 ARQ 기법으로 인한 ACK storm을 해결하기 위해 많은 연구들이 이뤄지고 있다.

DRME (dynamic multicast retransmission encoding)와 CMRE (cache-based multicast retransmission encoding)[2]는 소스 노드의 코딩 기회를 제어해서 재전송 과부하를 줄이는 방법이다. DMRE는 수신 노드들의 요구가 가장 많은 패킷을 인코딩하여 재전송하는 기법으로 송신 노드의 인코딩 결정을 수신 노드들의 요구에 의해서 결정하는 방법이다. DMRE에서 수신 노드는 코드화 패킷을 저장할 버퍼를 갖고 있지 않다. CMRE는 수신 노드가 디코딩에 성공할 때까지 수신된 코드화 패킷을 모두 버퍼에 저장한다. 두 기법은 XOR 코딩을 기반으로 한다.

무선 센서 네트워크 환경을 기반으로 하는 AdapCode[12]는 무선 센서 네트워크의 특성을 고려하여 네트워크 내의 모든 노드들은 인코딩과 디코딩을 수행한다. 각 노드의 코딩 계수 N은 이웃 노드들의 수에서 결정된다. 송신 노드와 이웃 노드들 간에 NACK 기법이 적용된다. N개의 패킷을 수신하지 못한 이웃 노드는 NACK를 전송하여 추가의 패킷을 수신할 수 있다.

R-Code와 GreedyCode는 무선 매쉬 네트워크를 기반한다. R-Code[11]는 100%의 패킷 전송률을 보장하기 위해 코드화 패킷의 효율적인 분산하기 위한 방법을 제안한다. 모든 노드들의 reliable reception을 보장하기 위해 이웃하는 노드들 간에 guardian-ward 관계를 형성하는 방법이다. guardian은 일시적 소스 노드 역할을 하게 되어 소스 노드의 송신 부담을 줄일 수 있다. R-Code는 guardian들의 중복된 패킷 전송을 차단함으로써 불필요한 트래픽 부하를 15% 줄였고 guardian-ward 관계에서 ACK 사용으로 브로드캐스트 지연시간이 65%로 줄어 들었다. GreedyCode[13]는 트리 구조를 사용하지 않고 중간 노드에 선택적 전달 기법을 도입하여 데이터 처리율을 높였다.

CodeCast[14]는 애드-혹 멀티캐스트 네트워크를 기반으로 하는 네트워크 코딩 프로토콜로써 패킷 손실 회복과 제어를 목적으로 한다. CodeCast에서는 패킷 손실 회복과 제어를 위해서 블록 기반 전송 기법을 도입하였다. 블록 기반 전송 기법은 하나의 파일을 여러 블록으로 나눈 후에, 송신 노드와 수신 노드 간에 블록 단위로 패킷들의 송수신 상태를 확인하는 것이다. 수신 노드는 블록타임아웃(block timeout)을 적용하여 블록을 수신한다. CodeCast에서 소스 노드는 파일은 동일한 크기의 블록으로 나눈다. 블록의 크기는 미리 정해진 값이고 모든 네트워크 코딩 노드에서 동일하기 적용된다. 각각의 블록들은 고유번호로 구분되며, 각 블록에는 블록타임아웃이 설정되어 있다. 블록타임아웃은 모든 블록에 동일하게 적용된다. 네트워크 코딩 노드는 블록타임아웃 이전에 블록을 수신하고, 블록타임아웃 이후에는 블록 수신 완료와 관계없이 네트워크 코딩과 전송을 수행한다.

CodeCast의 블록 단위 전송 기법과 *r*NC의 코드화 그룹 단위 전송 기법은 신뢰성을 기반으로 하고 있다는 점에서 유사하다. 그러나, CodeCast는 무선 애드-혹 네트워크를 위한 네트워크 코딩 프로토콜이고 rNC은 유무선 네트워크에 포괄적으로 적용 가능한 기법이다. CodeCast에서 블록 사이즈는 고정된 값이지만, *r*NC에서 CPG는 각 네트워크 코딩 노드의 입력 트래픽 부하와 네트워크 코딩 서비스 방법에 따라 크기가 달라 질 수 있다.

Ⅲ. 신뢰성 있는 그룹 기반 전송 모델

3.1. 네트워크 모델

본 논문에서는 홉 간의 패킷 소실이 발생할 수 있는 비신뢰성 네트워크를 고려하고 있다. 본 논문에서 네트워크의 노드들은 소스 노드, 네트워크 코딩 노드, 전달 노드 그리고 목적지 노드로 구분된다. 각 노드들에 대한 정의는 [15]에 따른다. 소스 노드는 파일 단위로 패킷들을 발생시킨다. 소스 노드는 한 파일로부터 둘 이상의 코딩 패킷 그룹들을 생성하고 네트워크 코딩 패킷 그룹 송수신 기법에 따라 전송한다. 네트워크 코딩 노드들은 네트워크 코딩을 수행하는 중간 노드로서 수신된 패킷들에 대한 디코딩은 수행하지 않는다. 전달 노드는 네트워크 코딩과 디코딩을 수행하지 않고 패킷들을 단순 전달하는 노드다. 목적지 노드는 소스 노드에서 발생된 파일을 수신하고 디코딩을 수행하는 노드이다.

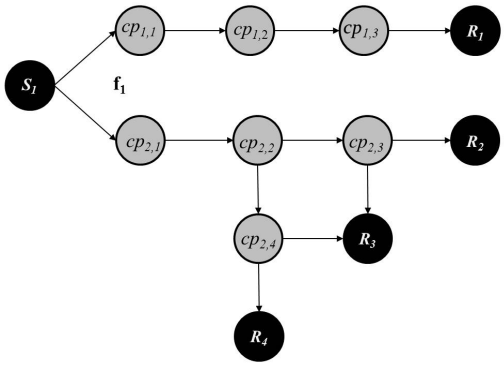

그림 1. *r*NC 프로토콜을 적용한 네트워크 코딩 네트워크
Fig. 1. Network coding network with *r*NC

그림 1은 *r*NC를 적용하여 소스 노드 S_1와 목적지 노드들 R_1, R_2, R_3, R_4 사이의 관계를 방향 그래프(directed graph)로 나타낸 것이다. R_1, R_2, R_3, R_4는 소스 노드 S_1의 목적지 노드이다. 소스 노드 S_1는 파일 $\mathbf{f}_1$를 목적지 노드 R_1, R_2, R_3, R_4에 전달하는 경로를 나타낸 것이다. $cp_{1,1}$, $cp_{1,2}$, $cp_{1,3}$그리고 $cp_{2,1}$, $cp_{2,2}$, $cp_{2,3}$, $cp_{2,4}$은 네트워크 코딩 노드들이다. 소스 노드 S_1와 목적지 노드 R_1, R_2, R_3, R_4들 간에는 동일한 네트워크 코딩 노드를 부분적으로 공유하거나 서로 상이한 네트워크 코딩 노드들이 존재할 수 있다. 송수신 네트워크 코딩 노드들 사이에는 다수의 전달 노드가 존재할 수 있는데 그림 1에서는 생략되어 있다.

멀티-홉 네트워크 코딩이 적용된 네트워크에는 소스 노드와 목적지 노드 간에 하나 이상의 네트워크 코딩 노드들이 존재한다. 소스 노드에서 발생된 파일의 패킷들은 네트워크 코딩 노드들을 거치면서 반복적으로 코딩이 이뤄지고 목적지 노드에 전달된다.

3.2. 큐 모델과 CPG 결정 알고리즘

*r*NC에서 노드들은 하나 이상의 입력 큐와 출력 큐를 갖는 폴링 시스템이라고 가정한다. 각 네트워크 코딩 노드의 폴링 시스템은 GSD (gated service discipline)에 따른다[16]. 네트워크 코딩을 수행하기 위해서는 두 개 이상의 패킷들이 큐에 누적되어야 하므로 이런 요구사항을 큐 모델에서부터 제공할 수 있도록 하기 위해 폴링 시스템을 가정한다. 본 논문에서는 입력 큐들간의 스위칭 오버 시간을 0이 아닌 상수 값으로 가정한다. 각 노드에서 네트워크 코딩 기법은 랜덤 선형 네트워크 코딩[17] 방식에 따른다.

소스 노드는 하나의 입력 큐와 복수개의 출력 큐를 갖는다. CPG 별로 출력 큐가 생성되고, 전송이 완료된 CPG의 출력 큐는 삭제된다. 둘 이상의 출력 큐가 존재할 때, 패킷들은 출력 큐 단위로 라운드 로빈 방식으로 전송된다. 그림 2의 예에서 소스 노드는 총 24개의 패킷으로 나누어 전송할 수 있는 파일에 대해 3개의 CPG 즉, $cpg_{1,1}$, $cpg_{1,2}$, $cpg_{1,3}$을 사용한다. 소스 노드는 $cpg_{1,1}$의 전송이 완료될 때까지 $cpg_{1,1}$에 대응하는 출력 큐에 있는 패킷들을 랜덤 선형 네트워크 코딩 기법으로 조합하여 전송한다. $cpg_{1,1}$의 전송이 완료되면 $cpg_{1,2}$의 패킷들에 대해 $cpg_{1,1}$의 패킷들과 동일한 절차에 따라 전송을 진행한다.

전달 노드는 하나의 입력 큐와 하나의 출력 큐를 갖는다. 최종 목적지 노드는 복수개의 입력 큐와 이들을 각각을 위한 디코딩 버퍼를 갖는다.

네트워크 코딩 노드는 복수 개의 입력 큐와 복수개의 출력 큐를 갖는다. 입력 큐의 개수는 코딩 노드가 담당하는 세션의 수에 의해 결정된다. 네트워크 코딩 노드에서 수신한 패킷들은 세션 별로 입력 큐에 저장된다. 새롭지 않은(non-innovative) 코딩 벡터를 갖는 패킷은 입력 큐에 저장되지 않고 삭제된다. 네트워크 코딩 노드는 입력 큐에 쌓인 패킷들에 대해 GSD 폴링 원칙에 따라 한 번에 획득되는 패킷들을 하나의 CPG로 정의한다. 즉, *r*NC에서

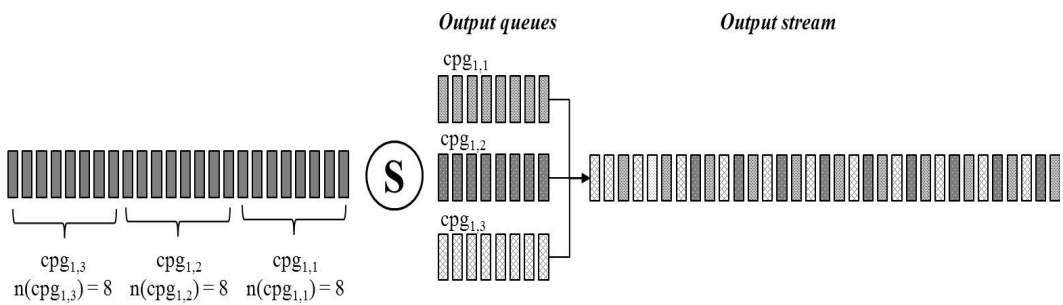

그림 2. 네트워크 코딩 노드 - 파일의 크기가 24인 소스 노드의 간단한 네트워크 코딩 예
Fig. 2. Network coding node - example of source node with file size 24

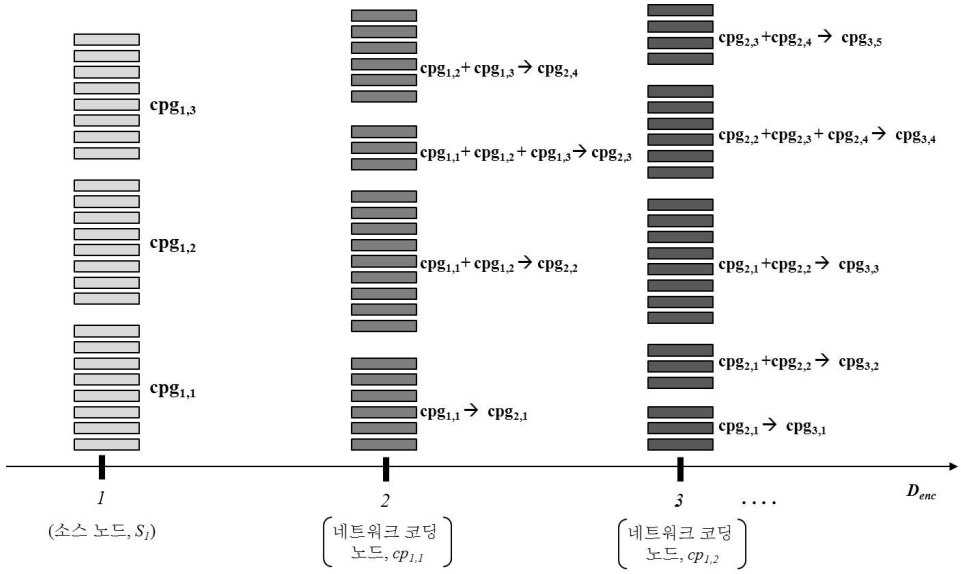

그림 3. 동적 CPG 생성 사례
Fig. 3. Dynamic CPG formation example

CPG에 속하는 패킷의 수는 고정된 값이 아니고, 입력 패킷 부하 정도, 세션의 수, 코딩 서비스 주기 등에 의해 달라질 수 있다. 네트워크 코딩 노드는 각 출력 큐에 속한 패킷들을 인코딩하여 전송한다. CPG 별로 출력 큐가 만들어지며, 출력 큐에 대한 처리 과정은 소스 노드와 동일하다.

그림 3은 그림 1의 소스 노드(S_1)와 네트워크 코딩 노드들($cp_{1,1}$, $cp_{1,2}$, $cp_{1,3}$)의 CPG 결정 과정을 나타낸다. 소스 노드 S_1과 가장 인접한 네트워크 코딩 노드 $cp_{1,1}$ 간의 CPG 기반 패킷 전송 과정을 설명한다. 소스 노드 S_1은 전송하고자 하는 파일로부터 3개의 CPG, $cpg_{1,1}$, $cpg_{1,2}$, $cpg_{1,3}$을 결정하고 각 CPG에 대해 출력 큐를 생성하고 각 출력 큐에 속한 패킷들에 대해 조합하여 전송한다. 이 때, CPG 단위의 ARQ를 수행할 수 있도록, 전송되는 패킷들에 대해 CPG 고유번호를 기록하고, 해당 CPG에 속한 패킷의 수를 기록한다. 소스 노드로부터 가장 인접한 네트워크 코딩 노드 $cp_{1,1}$는 소스 노드 S_1으로부터 $cpg_{1,1}$에 속한 조합된 패킷들을 수신하여 입력 큐에 저장된다. 이 때, 기존의 입력 큐에 저장된 동일한 CPG에 속한 패킷들과 비교하여 새롭지 않은 패킷들은 버린다. $cp_{1,1}$의 입력 큐 폴링 원칙인 GSD에 따라 한 번에 획득된 패킷들에 대해 새로운 CPG인 $cpg_{2,1}$을 결정한다. 즉, $cpg_{1,1}$에 속한 패킷들을 코딩하여 만들어진 패킷들이 $cp_{1,1}$에서는 서로 다른 CPG에 속할 수 있다.

3.3. CPG 기반 데이터 송수신

본 논문에서는 네트워크 상에 동시에 여러 소스 노드가 다수의 목적지 노드로 데이터를 전송하는 경우를 고려한다. 따라서, 한 CPG 송신 노드에서 전송하는 CPG의 수신자가 둘 이상일 수 있고, 하나의 CPG 수신 노드는 하나 이상의 CPG 송신 노드들로부터 패킷을 수신할 수 있다.

그림 4는 복수 개의 멀티캐스트 세션들이 존재하는 네트워크에서 단일-홉 네트워크 코딩 노드들 간의 CPG 송수신 관계를 나타내기 위한 네트워크 CPG 송수신 모델을 나타낸다. 실선은 CPG 수신 노드 y_j의 수신 범위를 나타내고 점선은 CPG 송신 노드 x_i의 송신 범위를 나타낸다. 즉, CPG 수신 노드 y_j는 CPG 송신 노드 x_1, …; x_i, …; x_m 로부터 패킷들을 수신하고, CPG 송신 노드 x_i는 CPG 수신 노드 y_1, …; y_j, …; y_n 에게 패킷들을 송신함을 나타낸다.

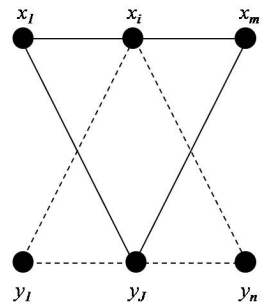

그림 4. 원-홉 네트워크 코딩의 CPG 송수신 모델
Fig. 4. CPG exchange model of one-hop network coding

그림 4에서 x_i ($i=1,\cdots,m$)는 CPG 송신 노드이고, y_j ($j=1,\cdots,n$)은 CPG 수신 노드이다. 송신 노드 x_i는 수신된 k개의 패킷들을 랜덤 선형 코딩 기법 등으로 코딩하여 전송한다. 이 하나의 코딩 패킷을 생성하기 위해 묶여진 k개의 패킷들을 하나의 CPG이라 칭한다. 멀티캐스트를 가정하므로, x_i가 송신한 패킷들은 하나 이상의 CPG 수신 노드들에게 전달된다. CPG 송신 노드는, 예상하는 CPG 수신 노드들이 모두 해당 CPG를 수신 완료했음을 확인할 때까지, CPG를 형성하는 k개의 패킷들을 사용해서 서로 다른 코딩 벡터를 사용하여 이전에 전송한 패킷과 독립적인 코딩 패킷들을 지속적으로 생성하여 전송한다.

CPG 송신 노드와 CPG 수신 노드는 CPG 단위로 오류 복구 기능을 수행한다. CPG 송신 노드는 CPG 수신 노드로부터 수신 완료 메시지를 수신할 때까지 서로 다른 네트워크 코딩 벡터를 적용하여 독립적인 코딩된 패킷을 만들어낼 수 있다. 따라서 CPG 수신 노드는 중복되는 패킷을 수신하는 경우가 없고, 수신된 모든 패킷들을 활용하여 새로운 CPG를 생성할 수 있다.

소스 노드와 최종 목적지 노드, 소스 노드와 소스 노드로 부터 첫번째 네트워크 코딩 노드, 송수신 네트워크 코딩 노드, 소스 노드로부터 마지막 네트워크 코딩 노드와 최종 목적지 노드 등의 관계는 그림 4의 네트워크 CPG 송수신 모델로 나타낼 수 있다. 소스 노드와 목적지 노드의 네트워크 CPG 송수신 모델에서 CPG는 소스 노드에서 발생시킨 파일에 해당된다. 따라서 목적지 노드는 새로운 파일의 패킷을 처음 수신했을 때, 이를 소스 노드에게 알린다. 그리고 해당 파일을 모두 수신하고 디코딩에 성공했을 때, 이를 소스 노드에게 알린다. 소스 노드는 모든 목적지 노드들로부터 파일 수신 완료를 확인한 후, 새로운 파일을 발생시킬 수 있다. 소스 노드로부터 마지막 네트워크 코딩 노드와 목적지 노드의 네트워크 코딩 패킷 그룹 송수신 모델에서 목적지 노드는 그림 3의 수신 노드 y_j와 같다.

CPG 수신 노드 y_j는 새로운 CPG를 수신하게 되면, CPG 송신 노드에게 이를 통지한다. 그리고, 해당 CPG의 수신이 완료되면 또한 이를 CPG 송신 노드에게 통지한다. CPG 수신 노드는 자신이 목적지 노드가 아니므로 수신한 패킷들을 사용해서 다음 CPG 수신 노드에게 코딩된 패킷을 전송한다. 이 때, 해당 CPG의 패킷들을 모두 수신하지 않더라도 일부의 패킷들을 사용해서 코딩을 적용할 수 있다. CPG 결정에 대한 알고리즘은 III의 3절에 제시되어 있다.

```
procedure NCP_ENCODE
  REPEAT
    Take some packets from the input queues according to the polling service discipline
    Encode the packets
    Create the coded packet group information
    Keep the coded packet group information in this node
    Payload the coded packet group information onto the header of all packets in the coding group
    Enque the packets onto the output queue for the coded packet group
  UNTIL the input queues are empty


procedure NCP_SEND
  REPEAT
    Deque a packet from the output queue for a coding group
    Send the packet
  UNTIL all the receivers receive the coding group
```

그림 5. CPG 송신 노드의 처리 알고리즘
Fig. 5. Algorithm for CPG sender

그림 5에서 NCP_ENCODE()는 CPG 송신 노드의 인코딩 과정를 나타내고 NCP_SEND()는 CPG 송신 과정을 나타낸다. CPG 송신 노드는 입력 큐에 있는 패킷들로 CPG를 생성한다. CPG 생성시에 CPG 정보가 동시에 생성된다. CPG 정보는 이 송신 노드에 저장되어 CPG의 송수신을 관리하는데 사용된다. CPG 정보에는 CPG 고유번호, CPG 크기, CPG 송신 노드 고유번호 등이 있다. CPG의 모든 패킷들의 헤더에 CPG 정보가 실린다. 결정된 CPG의 패킷들은 해당 출력 큐에 넣는다. CPG 수신 노드들이 해당 CPG의 패킷들을 모두 수신할 때까지 즉, CPG 수신 노드들이 CPG 수신 완료 메시지를 전송할 때까지 인코딩 동작은 반복된다. 그림 6에서 NCP_SEND()는 CPG 수신 노드가 출력 큐들의 패킷들을 주기적으로 꺼내어 전송하는 과정을 나타낸다. CPG 송신 노드은 출력 큐들이 모두 빈 상태가 될 때까지 이를 반복한다.

```
procedure NCP_RECEIVE
  IF a packet is received and the coded packet group id of the packet is new
    Notify to the sender of the packet
    Record the coded packet group information
    Record the coding vector of the packet
  ELSE
    Check whether a packet is duplicated or not
    Check whether the vector of a packet is innovative or not
    IF the vector of the packet is innovative and not duplicated
      Record the vector of the packet
      Enque the packet onto the input queue
    ELSE
      drop the packet
    ENDIF
  ENDIF
```

그림 6. CPG 수신 노드의 처리 알고리즘
Fig. 6. Algorithm for CPG Receiver

그림 6에서 NCP_RECEIVE()는 CPG 수신 노드의 패킷 수신 과정을 나타낸다. 수신된 패킷이 수신 완료된 CPG의 패킷인지 확인한다. 이미 수신 완료된 CPG의 패킷은 차단된다. 만일, 새로운 CPG의 패킷이라면 해당 CPG 송신 노드에게 CPG 수신 노드로 등록한다. 수신된 패킷이 수신 중인 CPG의 패킷이라면 코딩 벡터의 혁신성을 확인한다. 중복 수신된 패킷과 코딩 벡터가 혁신적이지 않은 패킷은 차단 된다.

Ⅳ. 성능 분석

4.1. 시뮬레이션 환경

본 논문에서 시뮬레이터 ns-2에 *r*NC를 모듈을 추가하였다. 특정 라우팅 알고리즘을 가정하지 않고 설계된 기법이므로, 최종 수신 노드가 소스 노드에게 그룹 참여를 요청하는 메시지를 전송하는 절차만을 진행하고, 이로 인한 경로 설정을 가정하지 않는다. 노드들은 플러딩 기법으로 이웃 노드들에게 패킷을 전달한다.

이동성이 없는 무선 네트워크에 100개의 노드들을 배치하였다. 각 노드의 신호 범위 내에 평균적으로 4개의 노드가 배치되는 그리드 형태의 네트워크를 구성하였다. 소스 노드는 1~2개고, 소스 노드와 최종 수신 노드들 사이에 존재하는 네트워크 코딩 노드는 평균적으로 30 ~ 50 개이다. 하나의 소스 노드당 2 ~ 10 개의 최종 수신 노드들이 있다. 네트워크의 링크 당 패킷 손실률(*Erasure rate*)은 0.0 ~ 0.25 이다.

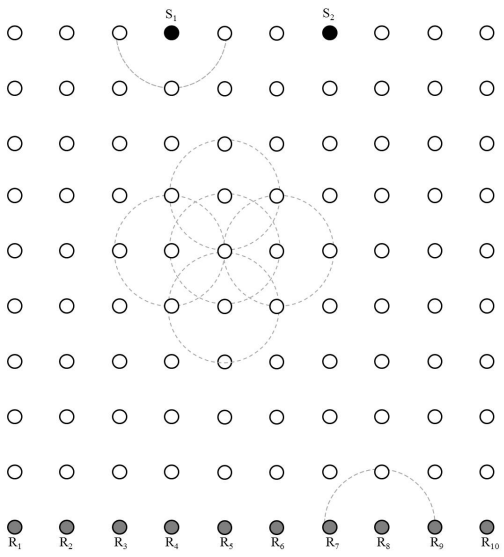

그림 7. 시뮬레이션 네트워크 토폴로지
Fig. 7. Network topology for si,mulation

4.2. 네트워크 코딩 지연 시간

네트워크 코딩 지연 시간은 파일의 첫 패킷이 소스 노드에서 전송된 시점부터 소스 노드가 목적지 노드들의 디코딩 완료 사실을 모두 인지한 시점까지의 시간이다. 목적지 노드가 2개 이상일 때, 소스 노드는 가장 마지막에 수신된 파일 수신 완료메시지를 기준으로 네트워크 코딩 지연 시간을 측정한다.

기존의 패킷 전송 기법은 소스 노드가 첫번째 패킷을 전송하면 그 패킷이 변형 없이 목적지 노드에 전달되므로 개별 패킷에 대해 양단간 지연 시간(end-to-end delay)를 측정하는 것이 가능했다. 네트워크 코딩을 적용함으로써, 각 패킷이 타 패킷들과 조합하여 전송되므로 목적지 노드가 다수의 패킷들을 수신하여 디코딩해야만 원본 패킷의 성공적인 수신을 확인할 수 있다. 즉, 네트워크 코딩을 적용할 시에는 패킷 당 양단간 지연 시간을 측정하는 것이 불가능하므로, '네트워크 코딩 지연 시간(network coding delay)'이라는 새로운 측정치를 정의하였다.

그림 8은 소스 노드 수가 2이고, 각 소스 노드에 대한 목적지 노드들이 각 3개 있다. 링크 당 패킷 손실률은 무선 네트워크의 각 홉에 적용 되었다. 소스 노드에서 전송한 파일의 크기 f_n을 16에서 512 패킷로 변화시켜 가면서 네트워크 코딩 지연 시간을 측정하였다. 이미 예상할 수 있는 대로, 링크의 에러율이 높아질수록 네트워크 코딩 지연 시간은 파일 크기에 무관하게 증가하는 것을 확인할 수 있다.

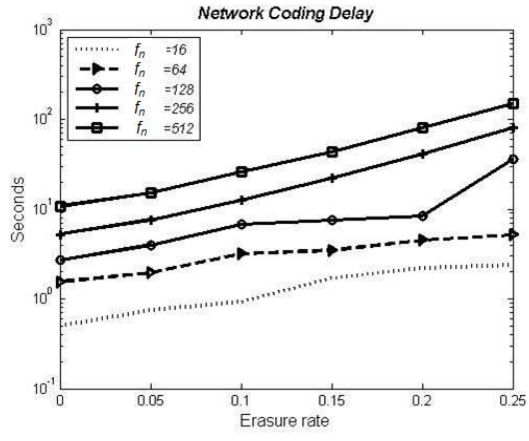

그림 8. 링크의 에러율에 따른 파일 크기 별 네트워크 코딩 시간의 변화
Fig. 8. Network coding delay as to file size according to the link erasure rate

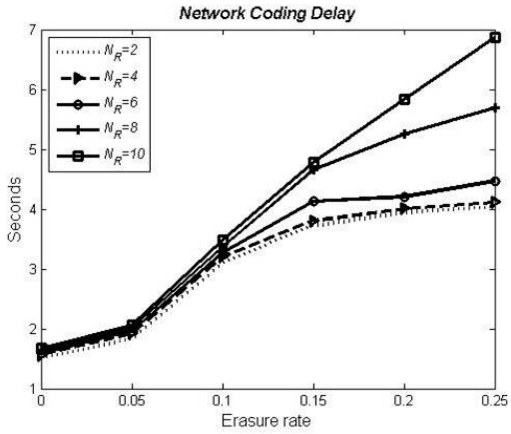

그림 9. 링크 패킷 손실률에 따른 수신 노드 개수 별 네트워크 코딩 지연 시간의 변화 (f_n=64)
Fig. 9. Network coding delay as to the number of receivers according to the packet loss rate of link

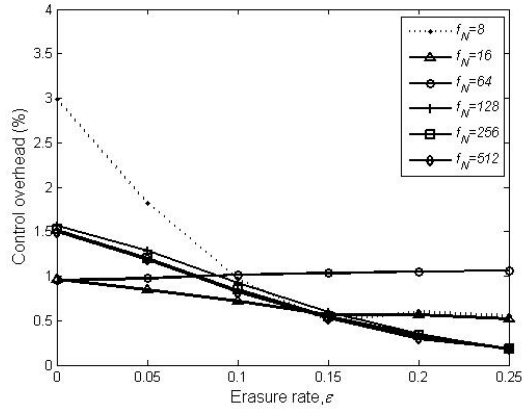

그림 10. 링크의 에러율에 따른 파일 크기 별 제어 부담률의 변화
Fig. 10. Control overhead as to file size according to the link erasure rate

그림 9는 소스 노드 수가 2이고, 각 소스 노드 당 목적지 노드의 수 N_R을 2에서 10으로 변화시켜 가면서 네트워크 코딩 지연시간을 측정한 결과를 보여주고 있다. N_R이 커질수록 링크 패킷 손실률에 증가에 따른 네트워크 코딩 지연 시간의 증가 정도가 커지는 것을 볼 수 있다. 그림 9에서 N_R이 10일 때 네트워크 코딩 지연 시간과 링크의 패킷 손실률이 비례관계에 있지만, N_R이 2일 때는 비례관계에 있지 않음을 볼 수 있다. 네트워크 내의 노드의 수가 일정할 때, N_R의 증가는 네트워크 코딩과 패킷 전달에 참여하는 노드들의 수의 감소를 초래하게 되고, 이는 패킷 수신률 감소를 초래하게 된다[18]. 즉, N_R이 10일 때는 N_R이 2일 때보다 네트워크 코딩과 패킷 전달을 하는 노드들의 수가 8개 적다. 따라서, N_R이 10일 때는 N_R이 2일 때보다 패킷 수신률이 떨어지게 되어 네트워크 코딩 지연시간의 증가폭이 커지게 된다.

4.3. 제어 부담률 (Control overhead)

제어 부담률은 전체 네트워크에서 전송된 메시지 중에서 데이터를 제외하고 전달 기법의 제어를 위해 전송된 메시지의 비율을 의미한다. *r*NC에서는 파일 수신자 등록을 위한 메시지, 목적지 노드가 소스 노드에게 파일 수신 완료를 알리기 위한 메시지, CPG 수신자 등록을 위한 메시지, 그리고 CPG 수신 완료를 알리기 위한 메시지 등이 제어를 위해 1 전송된 메시지에 해당한다.

그림 10은 소스 노드의 수가 2이고, 각 소스 노드에 등록된 최종 수신 노드의 수가 3일 때, 제어 부담률을 측정한 값이다. 패킷의 크기가 일정할 때, 링크의 패킷 손실률을 0.0에서 0.25로 변화시켜 가면서, 전송하고자 하는 파일을 구성하는 패킷의 수 f_n이 8, 16, 64, 128, 256, 512 일 때 제어 부담률을 백분율로 측정하였다. 제어부담률은 모든 패킷 손실률에 대해 최대 3% 이하를 유지하고 있다. 패킷 손실률이 증가함에 따라 제어 부담률이 일정한 수준을 유지하거나 낮아지는 점이다. 즉, 일반적으로 링크의 패킷 손실률이 증가할수록 데이터 전송 회수는 증가하므로 제어 부담률은 상대적으로 감소하게 된다.

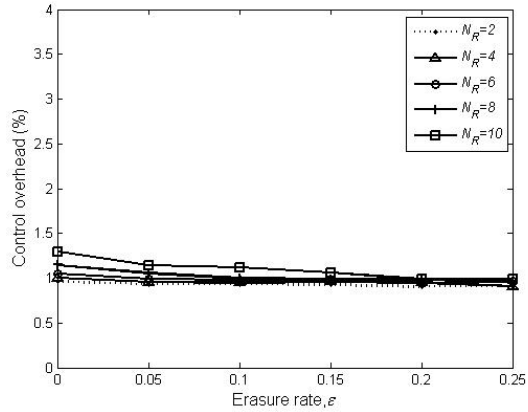

그림 11. 링크의 에러율에 따른 최종 수신 노드 개수 별 제어 부담률의 변화
Fig. 11. Control overhead as to the number of receivers according to link erasure rate

그림 11은 파일 크기 f_n가 64일 때, 소스 노드 수를 2로 고정하고, 링크의 패킷 손실율을 0.0에서 0.25까지 증가시켜 가면서, 각 소스 노드의 최종 수

신 노드의 수 N_R가 2, 4, 6, 8, 10 일 때 제어 부담률의 변화를 백분율로 보여주고 있다.

종합적으로 그림 10과 11을 통해 *r*NC의 제어 부담률은 파일의 크기와 최종 수신 노드 수에 큰 영향을 미치지 않음을 알 수 있다. 일반적인 ARQ 방법에서 파일 크기가 커지거나 최종 수신 노드 수가 증가하면 제어 부담률이 증가한다. 그러나 *r*NC에서는 CPG 단위 송수신 기법을 적용하여 파일 크기가 증가해도 제어 부담률은 일정한 값을 유지한다.

4.4. *r*NC와 CodeCast의 성능 비교

CodeCast[14]는 단일 멀티캐스트 세션만을 고려한 프로토콜이어서, *r*NC와 CodeCast 모두 소스 노드의 수를 1로 고정하여, 네트워크 코딩 지연 시간이라는 관점에서 성능을 비교하였다. f_n=64로 고정하였고, 링크의 패킷 손실률을 0.0에서 0.25로 변화시키면서, 수신 노드의 수를 1, 3, 7로 변화시켜 가면서 성능을 관찰하였다.

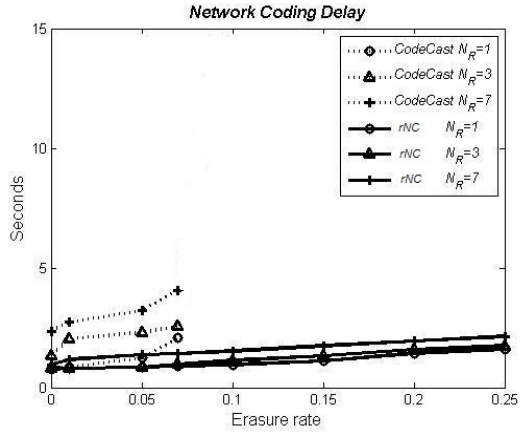

그림 12. 링크의 패킷 손실률 변화에 따른 *r*NC와 CodeCast의 네트워크 코딩 지연 시간 비교
Fig. 12. Network coding delay of *r*NC and Codecast according to the link erasure rate

그림 12에서 볼 수 있는 바와 같이, 전체적으로 CodeCast의 네트워크 코딩 지연시간이 *r*NC 프로토콜과 네트워크 코딩 지연 시간보다 크다. 특히, CodeCast에서는 Erasure rate이 0.07이상이 되면 목적지 노드는 디코딩을 수행할 충분한 패킷들을 수신하지 못하여 디코딩에 실패하였고 네트워크 코딩 지연시간이 측정되지 않았다. 이는 해당 시뮬레이션 시간 동안 목적지 노드들이 원본 파일을 복원하기에 필요한 개수의 혁신적인 패킷들을 수신하지 못했기 때문이다.

CodeCast는 패킷 조합을 위해 block timeout 동안 동일 블록 내의 패킷들을 모으고, block timeout 이후에는 수집된 패킷들에 대해 조합해서 다음 노드에게 전달한다. 해당 블록의 원본 패킷들을 복원하기 위해 필요한 수의 패킷들이 수신되지 못했다면, 이웃 노드들이 자발적으로 보유한 패킷들 중에서 적절한 패킷을 재전송해 주는 기법을 사용하고 있다. CodeCast는 소스 노드로부터 목적지 노드까지 블록 단위의 전송을 관리하는 기법으로, 개별 블록마다 소스 노드부터 목적지 노드까지 원본 패킷 복원에 필요한 수의 패킷 전송이 보장되는 경우도 있지만, 그렇지 못한 경우도 발생할 수 있다. 즉, 파일을 하나의 블록으로 정의하지 않는 한 CodeCast는 파일의 복원에 필요한 패킷들의 전송을 보장할 수 없다.

*r*NC는 파일 단위로 소스 노드부터 목적지 노드까지 전달을 보장하기 위한 기법으로, 각 네트워크 코딩 노드들 간에 ARQ를 적용함으로써 CodeCast에 비해 높은 수준의 신뢰성을 보장할 수 있으므로 모든 목적지 노드가 원본 파일을 복원하기에 필요한 충분한 수의 조합된 패킷들을 수신할 수 있었다.

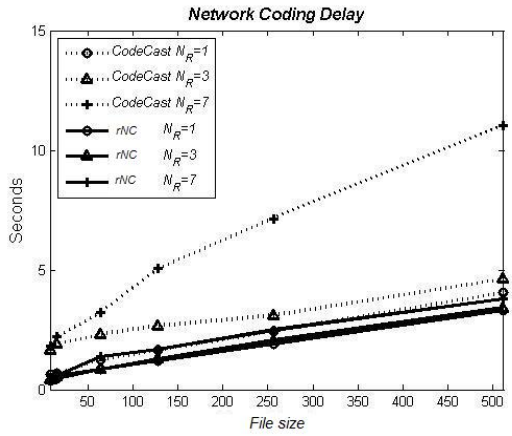

그림 13. 발생 파일크기에 따른 *r*NC 프로토콜과 CodeCast의 네트워크 코딩 지연 시간 비교
Fig. 13. Network coding delay vs. the file size

그림 13에서 패킷 손실률을 0.05로 고정시키고, f_n을 8에서 512 패킷으로 증가시켜 가면서 네트워크 코딩 지연 시간을 측정한 결과를 보여주고 있다. 이 때, 목적지 노드의 수를 1, 3, 7로 변화시켜 가면서 성능을 측정하였다.

CodeCast의 네트워크 코딩 지연시간은 f_n의 증가에 큰 영향을 받는다. 이와는 상대적으로 *r*NC의 네

트워크 코딩 지연시간은 f_n의 증가에 따라 그 증가 폭이 작다. CodeCast는 소스 노드에서부터 코딩 패킷 그룹의 크기가 고정되어 있고 블록 단위로만 인코딩과 디코딩이 발생한다. 이는 CodeCast가 응용 프로그램의 전송 단위가 정해져 있는 상태에서 이를 네트워크 코딩 단위로 대응하기 위한 정책이다. 그러므로 f_n이 증가하면, 블록의 수가 증가하게 되고, 블록 수에 비례하여 네트워크 코딩 지연 시간이 증가하게 된다. 그러나, *r*NC에서는 파일 단위로 소스 노드에서 목적지 노드로 전송을 완료하기 위한 기법으로 파일 내의 임의의 패킷들이 조합되는 것을 허용한다. 즉, CodeCast에 비해 파일을 구성하는 패킷의 수에 영향을 덜 받는다.

V. 결 론

본 논문에서는 이동성이 없는 다중 홉 무선 네트워크 환경에서 파일 전송을 위한 코딩 패킷 그룹 기반 신뢰성 있는 네트워크 코딩 (Coding Packet Group-based Reliable Network Coding, *r*NC) 기법을 제안한다. 기존의 기법들은 소스 노드에서 정의한 파일 단위로 코딩을 적용하고 신뢰성 있는 전달을 보장하는 방식이었다면, 본 논문에서 제안하는 기법은 각 네트워크 코딩 노드들이 폴링 시스템에 의해 일정 시점에 한 번에 획득되는 패킷들에 대해 코딩을 적용하고 인접한 네트워크 코딩 노드들 간에 신뢰성 있는 전달을 보장한다는 점에서 차이를 보인다.

CodeCast와의 성능 비교에서도 볼 수 있는 바와 같이, 블록 단위로 신뢰성 있는 전송을 보장하는 경우, 해당 블록의 전송이 마칠 때까지 다음 블록을 전송할 수 없어 전체적으로 데이터 전달 시간이 길어지게 된다. 시뮬레이션을 통한 성능 평가에서 관찰했듯이, 신뢰성 있는 데이터 전송을 위해서는 본 논문에서 제안하는 기법이 데이터 전달 시간이나 패킷 전달률에 있어서 기존에 제안된 기법보다 좋은 성능을 보일 수 있다.

본 논문에서 제안하는 기법이 다수의 입력 큐와 다수의 출력 큐를 필요로 하여 시스템 메모리를 다소 많이 요구하는 특징을 갖고 있으므로, 시스템 메모리를 보다 효율적으로 활용하기 위한 방안을 모색할 것이며, 최종 목적지 노드에서의 데이터 복원을 위한 디코딩 부담을 완화하기 위한 방안을 모색할 것이다.

References

[1] W. Yan, S. Yu, and Y. Cai, “Reliable multicast with network coding in lossy wireless networks,” *Int. J. Commun. Network Syst. Sci.*, vol. 3, no. 10, pp. 816-820, Oct. 2010.

[2] C. Zhan, Y. Xu, J. Wang, and V. Lee, “Reliable multicast in wireless networks using network coding,” in *Proc. IEEE 6th Int. Conf. Mobile Adhoc Sensor Syst. (MASS)*, pp. 506-515, Macau, China, Oct. 2009.

[3] A. Sobeih, H. Baraks, and A. Fahmy, “ReMHoc: a reliable multicast protocol for wireless mobile multihop ad hoc networks,” in *Proc. 1st IEEE Consumer Commun. Networking Conf. (CCNC)*, pp. 146-151, Las Vegas, U.S.A., Jan. 2004.

[4] E. Pagani and G. P. Rossi, “Reliable broadcast in mobile multihop packet networks,” MobiCom’97, in Proc. 3rd Annu. ACM/IEEE Int. Conf. Moblie Comput. Networking (MobiCom), pp.34-42, Budapest, Hungary, Sep. 1997.

[5] L. Rizzo and L. Vicisano, “RMDP: an FEC-based reliable multicast protocol for wireless environments,” ACM SIGMOBILE Mobile Comput. Commun. Review, vol. 2, no. 2, pp. 23-31, Apr. 1998.

[6] J. Nonnenmacher, E. Biersack, and D. Towsley, “Parity-based loss recovery for reliable multicast transmission,” IEEE/ACM Trans. Networking, vol. 6, no. 4, pp. 349-361, Aug. 1998.

[7] M. Ghaderi, D. Towsley, and J. Kurose, “Reliability gain of network coding in lossy wireless networks,” *Dept. Comput. Sci., Univ. Calgary, Technical Report:TR-07-08*, Jan. 2008.

[8] A. Erylmaz, A. Ozdaglar, and M. Médard, “On delay performance gains from network coding,” in *Proc. 40th Annu. Conf. Inform. Sci. Syst.*, pp. 864 – 870, Princeton, U.S.A., Mar.

2006.

[9] T. K. Dikaliotis, A. G. Dimakis, T. Ho, and M. Effros, "On the delay of network coding over line networks," in *Proc. IEEE Int. Conf. Symp. Inform. Theory (ISIT)*, vol. 2, pp. 1408-1412, Seoul, Korea, June-July 2009.

[10] A. Eryilmaz, A. Ozdaglar, M. Medard, and E. Ahmed, "On the delay and throughput gains of coding in unreliable networks," *IEEE Trans. Inform. Theory*, vol. 54, no. 12, pp. 5511-5524, Dec. 2008.

[11] Z. Yang, M. Li, and W. Lou, "R-Code: network coding-based reliable broadcast in wireless mesh networks," *Ad Hoc Networks*, vol. 9, no. 5, pp. 788-798, July 2011.

[12] I-H. Hou, Y.-E. Tsai, T. F. Abdelzaher, and I. Gupta, "AdapCode: adaptive coding for code updates in wireless sensor networks," in *Proc. IEEE INFOCOM 2008*, pp. 1517-1525, Phoenix, U.S.A., Apr. 2008.

[13] X. Tan, H. Yue, Y. Fang, and W. Cheng, "Greedy strategy for network coding based reliable broadcast in wireless mesh networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, pp. 5627-5632, Anaheim, U.S.A., Dec. 2012.

[14] J.-S. Park, M. Gerla, D. S. Lun, Y. Yi, and M. Medard, "Codecast: a network-coding-based ad hoc multicast protocol," *IEEE Wireless Commun.*, vol. 13, no. 5, pp. 76–81, Oct. 2006.

[15] Y.-S. Kim, K. Kang, and Y.-J. Cho, "On the network coding delay in multi-hop wireless networks with multiple coding points," in *Proc. 20th Int. Conf. Comput. Commun. Networks (ICCCN)*, pp. 1-5, Maui, U.S.A., July-Aug. 2011.

[16] O. J. Boxma, A. C. C. van Wijk, and I. J. B. F. Adan, "Polling systems with a gated/exhaustive discipline," in *Proc. 3rd Int. Conf. Performance Evaluation Methodologies and Tools (ValueTools)*, Article no. 41, Athens, Greece, Oct. 2008.

[17] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," IEEE Trans. Inform. Theory, vol. 52, no. 10, pp. 4413-4430, Oct. 2006.

[18] S. Omiwade, R. Zheng, and C. Hua, "Butteries in the Mesh: Lightweight localized wireless network coding," in *Proc. 4th Workshop Network Coding Theory Applicat. (NetCod)*, pp. 1-6, Hong Kong, China, Jan. 2008.

[19] T. J. Choi, K. Kang, Y.-J. Cho, and J.-H. Bang, " An inter-session opportunistic network coding-aware multipath routing protocol," *J. Korea Inform. Commun. Soc. (KICS)*, vol. 37, no. 7B, pp. 536-549, July 2012.

김 영 실 (Youngsil Kim)

1996년 2월 아주대학교 정보과학과 졸업
1998년 2월 아주대학교 정보과학과 석사
2005년 9월~현재 정보통신공학 박사과정
<관심분야> 멀티캐스트, 네트워크 성능분석 모델링, 네트워크 코딩

강 경 란 (Kyungran Kang)

1994년 2월 KAIST 석사
1999년 2월 KAIST 박사
2004년 3월~현재 아주대학교 부교수
<관심분야> 멀티캐스트, 이동 네트워크, 네트워크 코딩

조 영 종 (Young-Jong Cho)

1985년 2월 KAIST 석사
1990년 2월 KAIST 박사
1996년 3월~현재 아주대학교 교수
<관심분야> 멀티캐스트, 무선 네트워크, 트래픽 모델링, 네트워크 코딩