대형 멀티미디어 파일의 익명성 지원을 위한 수정 GNUnet

GNUnet improvement for anonymity supporting in large multimedia file

이 윤 진* 이 명 훈** 조 인 준***
Yoon-Jin, Lee Myoung-hoon Lee In-June Jo

요 약

GNUnet은 파일의 익명성 지원을 위해 파일을 1KB 블록크기로 분리하는 인코딩 방법을 채택하였고, 인코딩된 블록을 비구조적인 분산기법을 통해 피어들에게 분산시키고 있다. 하지만 이러한 인코딩 및 블록 분산 방안은 약 600~700MB의 대용량 멀티미디어 파일을 처리할 경우 첫째, 추가적으로 R블럭과 I블럭을 과도하게 요구하게 되어 약 4%의 저장공간을 낭비하는 문제를 지니고 있다. 둘째, 비구조적 브로드캐스팅 분산방법 때문에 네트워크 부하가 과도한 단점을 내포하고 있다.

본 논문에서는 이러한 문제점 해결을 위하여 인코딩 블록의 크기를 가변적으로 결정하는 방법과 구조적 위상을 통해 블록을 분산시키는 방안을 제안하였다. 제안 인코딩 방법은 추가 블록 생성을 1%이내로 최소화하였고, 분리된 블록의 분산을 위해 지정된 피어로 질의를 보내는 구조적 위상을 채택하여 블록 요청에서 발생하는 네트워크 부하를 감소시켰다.

Abstract

The GNUnet proposed a file encoding method by 1KB block size to support anonymity of files and decentralizes encoded block to peers through unstructured model. but, the encoding and block decentralizing method with 600~700MB large multimedia file appeared two problems. First problem, it need addition R block and I block, which make about 4% of storage resource. Second problem, unstructured model added network load by broadcasting decentralizing method.

This paper suggest variable encoding block size and structured model by block decentralizing solution. Suggested encoding method reduced block request supplementary block generation to 1% and network load by proposal structured model sending answer through dedicated peer to decentralize block

☞ Keyword : P2P, GNUnet, Chord, Anonymous

1. 서 론

최근 P2P 시스템의 공유자료는 최신 영화나 드라마와 같은 대형 멀티미디어 데이터가 중심을 이루고 있다. 그러나, 멀티미디어 데이터는 정보 제공자의 저작권 보호법에 의한 검열이 가능하기 때문에 자유로운 정보 공유를 위한 연구가 진행되었다. GNUnet은 자유로운 정보 공유를 위하여 익명성 지원방법을 제안한 시스템으로 파일 인코딩 방법과 인코딩된 블록을 분산하는 방법을 제안한 새로운 P2P 시스템이다.

GNUnet 시스템의 제안내용은 첫째, 원본 파일을 1KB의 크기 블록으로 나누어 D(Data)블록을 생성하고, D블록 탐색 및 지시를 위한 추가적인 블록인 I(Indirect), R(Root)블록을 생성한다. 대형 멀티미디어 데이터를 인코딩 하였을 경우 인코딩된 D블록을 지시하기 위한 R블록 과 I블록이 약 4%의 저장공간 낭비가 발생하였다. 둘째, GNUnet에서 제안된 블록 분산방법은 랜덤하게 2개의 근접노드를 선정하여 블록을 분산하는 방법을 제안

* 준 회 원 : 배재대학교 컴퓨터공학과 박사과정
gomyung@pcu.ac.kr
** 정 회 원 : 배재대학교 컴퓨터공학과 박사과정
lopez@mail.pcu.ac.kr
*** 정 회 원 : 배재대학교 컴퓨터공학과 교수
injune@mail.pcu.ac.kr
[2006/01/10 투고 - 2006/02/15 심사 - 2006/07/28 심사완료]

하였기 때문에 블록의 저장자는 피어를 경유할 때 마다 2n으로 증가되었고, 블록 탐색에 있어 과도한 네트워크 부하를 발생하는 브로드캐스트 방법이 수행된다.

따라서 본 논문은 GNUnet의 추가적인 저장공간 및 네트워크 부하 증가를 해결하기 위하여 인코딩 블록 크기를 "$1KB \leq 2^n \leq 1MB$" 범위에서 결정하는 가변적 블록크기 결정방법을 제안하였고, 분리된 블록을 분산을 위하여 Chord[7]에서 제안한 링형태의 구조적 모델을 새롭게 적용하였다. 이를 위하여 II장에서는 GNUnet의 인코딩 방법 및 블록 분산방법과 구조적 위상을 제안한 Chord에 대하여 설명하였다. III장에서는 이 논문에서 제시하는 성능개선 방법을 설명한다. IV장에서는 제안 시스템의 검토 및 고찰하였고, 마지막으로 결론을 맺었다.

2. 관련 연구

2.1 GNUnet 파일 인코딩 방법[1][2]

GNUnet은 익명성 지원을 위하여 파일을 1KB의 고정길이 블록으로 인코딩하는 방법을 제안하였다. 제안된 인코딩 방법은 파일의 내용을 암호화하고, 파일명을 해쉬값으로 하여 D(data)블록, I(Indirection)블록, R(root)블록을 생성한다. 그러나 대형 멀티미디어 데이터를 1KB 크기로 인코딩한 D블록을 지시하기 위하여 R블록과 I블록을 생성하였고, 추가된 블록은 원본 데이터의 4%에 해당하는 저장공간을 요구하였다.

2.2 GNUnet 인코딩 블록 분산 및 탐색방법

인코딩된 블록은 출판자의 익명성 제공을 위하여 다른 피어에게 블록을 저장한다. GNUnet에서 제안한 파일 분산 방법은 비구조적으로 구성된 피어의 라우팅 테이블을 기준으로 랜덤하게 피어를 선택하여 블록을 분산하고, 참고문헌[6]의 방식을 적용하여 TTL 값을 생성한다. 이러한 비 구조적방법의 블록 분배방법은 저장되는 피어를 랜덤하게 선택하기 때문에 블록이 저장된 피어가 어디인지 알 수 없다. 따라서 분산된 블록을 탐색하기 위하여 네트워크 부하를 증가시키는 브로드캐스트 방법을 사용하였다.

2.3 그누텔라 블록 분산 및 탐색방법

그누텔라의 특징은 P2P 비구조적 모델이라는 것이다. 모든 클라이언트는 서버이고, 모든 서버는 클라이언트이다. 일부 서번트들의 네트워크가 오프라인이 되더라도 전체 그누텔라 네트워크의 운영에는 지장을 받지 않는 분산적인 특징 때문에 서번트들의 네트워크는 매우 안정적인 장점을 지니고 있다. 그러나 많은 양의 패킷들이 개개인들의 연결을 유지하는데 소모됨으로써 네트워크 자원 소모가 크다는 단점을 갖고 있다. 즉, 네트워크 검색을 위하여 브로드캐스트에 의한 Ping/Pong 메시지가 전체 트래픽의 50% 이상을 차지한다. 또한 검색어가 전달되는 형태이기 때문에 대역폭이 느리거나 네트워크 상태가 좋지 않은 사용자들에 의해 병목 현상 발생하는 단점이 있다.

2.4 Chord 파일정보 분산 및 탐색 방법[7]

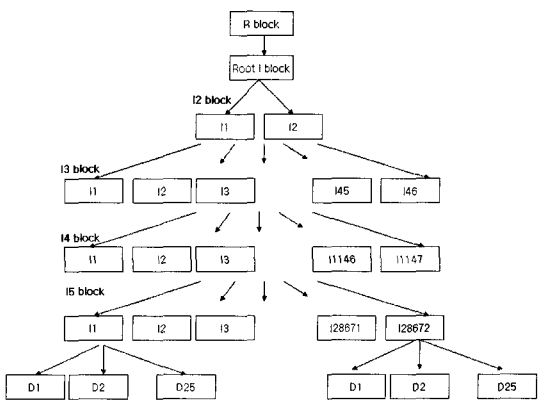
Chord 시스템은 P2P 응용을 위한 분산처리 자원 탐색 프로토콜이다. Chord 시스템은 피어의 위치를 링형의 구조적 모델로 구성하고, 피어 식별자를 기준으로 파일정보를 제공한다. 이러한 구조적 모델을 적용하기 위하여 피어식별자와 파일정보는 SHA-1 알고리즘과 같은 해쉬함수를 이용하여 해쉬되고, 파일정보의 해쉬 값을 피어 식별자와 비교하여 근접한 피어를 결정한다. 그러나 새로운 피어의 추가 및 탈퇴의 경우 정확한 파일정보를 유지하기 위하여 모든 피어가 피어 식별자 및 파일정보를 갱신해야 되기 때문에 과도한 네트워크 트래픽이 발생한다.

3. 제안 기법

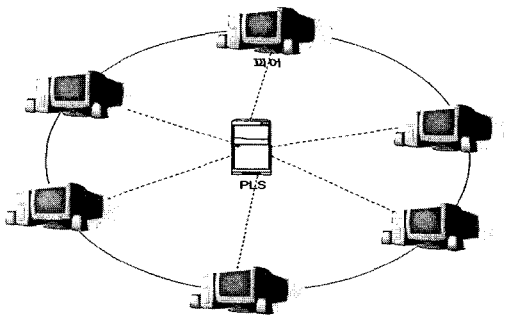
3.1. 시스템 구성

제안 시스템은 3가지의 익명성을 제공하기 위한 시스템이다. 첫 번째 정보제공자의 익명성, 두 번째 정보요청자의 익명성, 세 번째 정보저장자의 익명성이다. 따라서 3가지 익명성 제공을 위하여 원본파일의 인코딩하여 블록으로 분리하고, 분리된 블록을 제안 시스템 이용자에게 분산하는 방법을 제안하였다.

원본 파일 분리를 위한 인코딩과정을 수행할 경우 (그림 1)과 같은 트리형태로 구성된다. 트리형태는 3가지 블록으로 구성된다. 첫 번째, 원본파일을 분리한 D블록 생성하고, 두 번째, 분리된 D블록 지시를 위한 I블록을 생성한다. 마지막으로 키워드 검색을 위한 R블록을 생성한다. 인코딩과정에 의하여 생성된 블록들은 익명성 제공을 위하여 파일명과 데이터가 보호된다. 파일명은 160bit RIPE-MD[9]에 의하여 해쉬된 값이되고, 데이터는 제안 시스템에서 지정된 암호알고리즘에 의하여 암호화된다. 따라서 분리된 블록들은 익명성제공 및 동등한 저장공간 제공을 위하여 피어에게 분산되어야 한다. 제안 시스템은 분리된 블록 분산을 위한 피어의 구성은 (그림 2)와 같은 링형의 구조를 제안하였다.



(그림 1) 제안 시스템 인코딩된 블록



(그림 2) 제안시스템 위상

3.2 제안 시스템 인코딩 방법

제안 시스템은 대형 멀티미디어 파일의 공유시 발생하는 추가 공간을 최소화하기 위하여 인코딩 블록의 크기를 기존의 1KB에서 "$1KB \leq 2^n \leq 1MB$"의 범위인 가변크기 인코딩 방법을 제안하였다. GNUnet에서 제안한 1KByte의 블록 크기는 파일의 인코딩 및 블록 분산에서 최적의 성능을 제공할 수 있기 때문에 제안된 크기이다. 그러나 네트워크 환경 및 저장공간의 발전에 의하여 국내외 인터넷 이용자 PC 성능은 Giga단위의 저장공간과 100Mbps의 전송 속도를 제공하게 되었고, 1KByte의 인코딩 블록은 많은 수의 블록을 생성하기 때문에 네트워크 망에 과도한 트래픽을 제공하게 되었다. 따라서 현재 PC 및 네트워크 성능을 고려하여 "$1KB \leq 2^n \leq 1MB$" 최적의 범위를 결정하게 되었다.

(표 1) 용어 설명

기호	설명
F	원본 파일의 크기
BX	블록 크기와 가장 유사한 값
BC	결정된 인코딩 블록 크기
Bn	BC의 크기로 분리된 블록
n	대입되는 순차 값
Pn	피어를 식별하기 위한 피어 식별자
Bn	파일을 해쉬한 블록의 해쉬 값

(1) 인코딩 블록 크기 결정

GNUnet의 인코딩 방법은 1KB의 인코딩 블록 생성방법을 제안하였다. 그러나 1KB의 블록은 대형화된 멀티미디어 파일의 추가 블록을 생성하였고, 추가적인 저장공간을 요구하였다. 따라서 제안 시스템은 인코딩 블록의 크기를 "$1KB \leq 2^n \leq 1MB$" 의 범위에서 결정하였고, 결과적으로 1% 이내의 추가블록을 생성하였다. 다음은 인코딩 블록의 크기 결정 방법이다.

step1) GNUnet에서 제안된 I블록은 25개의 블록을 지시한다. 따라서 식 (1)과 같이 파일(F)을 25n으로 나누고, 나눈 값이 "$1KB \leq BX \leq 1MB$" 의 범위에 결과값을 구한다.

$$BX = F \div \sum_{n=1}^{25^n} \quad (1)$$

step2) 제안 시스템은 단편화 최적화를 위하여 2n의 크기로 구성된다. step1에서 결정된 블록의 크기는 2n의 값이 아니기 때문에 식 (2)의 조건을 만족하는 BC의 값을 구한다. BC의 범위는 "$1KB \leq BC \leq 1MB$" 이다.

$$BC = BX \leq \sum_{n=1}^{2^n} \quad (2)$$

step3) 결정된 인코딩 블록의 크기 BC에 따라 파일을 분리한다.

$$F = \{ B_1, B_2, \cdots B_n \} \quad (3)$$

(2) D블록 생성

결정된 BC의 크기에 따라 분리된 블록은 익명성 제공을 위하여 블록의 데이터를 암호화하고, 파일명은 160bit RIPE-MD로 해쉬한다. 다음은 D블록 생성 방법이다.

step1) 파일이 분리된 각 블록 Bi를 160비트 RIPE-MD로 해쉬한다.

$$\{H(B_1), H(B_2), \cdots H(B_n)\} \quad (4)$$

step2) 해쉬 값을 키로 하여 분리된 블록의 데이터를 암호화한다.

$$E_{H(D1)(B1)}, E_{H(D2)(B2)} \cdots E_{H(Dn)}(Bn) \quad (5)$$

step3) 생성된 블록을 160비트 RIPE-MD로 해쉬하여 D블록의 파일명을 결정한다.

$$H(E_{H(D1)}(D1)), \cdots H(E_{H(Dn)}(Dn)) \quad (6)$$

(3) I블록 생성

인코딩블록들은 다른 피어에게 분산되어 저장된다. 따라서 분산된 블록을 지시하기 위한 지시자가 필요하다. I블록은 분산된 D블록 또는 I블록을 지시하는 블록이다. 다음은 I블록 생성 방법이다.

step1) 총 25개의 D블록 또는 I블록에 대하여 식 (4)암호 키 값과 식 (6)의 블록 파일명을 저장한다.

$$H(D_1), H(E_{H(D1)}(D1)), (H(D_n), H(E_{H(Dn)}(Dn))) \quad (7)$$

step2) CRC32 값을 계산한다. (4B)

step3) 생성하고자 하는 I블록이 지시하는 25개의 블록 해쉬 값을 160비트로 해쉬한 20바이트의 슈퍼해쉬 값을 계산한다.

step4) D블록 또는 I블록을 지시하는 I블록을 생성한다.

$$(H(D_1), H(E_{H(D1)}(D_1))), (H(D_{25}), H(E_{H(D25)}(D_{25}))$$
$$(1000) + , CRC(4) + 슈퍼해쉬(20)4 \quad (8)$$

step5) D블록 생성방법의 step을 진행하여 블록을 암호화하고, 해쉬된 파일명을 생성한다.

(4) R블록 생성

D블록과 I블록은 파일명을 해쉬값으로 생성된

블록이다. 따라서 사용자 관점에서 탐색의 효율성을 제공하기 위하여 키워드에 의한 검색 방법을 제시하였다. R블록은 사용자의 키워드 검색을 제공하기 위하여 구성된 블록이다.

step1) 기밀 키워드 Ki를 선택하여 160비트 RIPE-MD로 해쉬하여 다음의 해쉬값을 계산한다.

$$H(K_i),\ H(H(K_i)) \quad (9)$$

step2) $H(K_i)$의 값을 키로 하여 루트 I블록의 파일명을 암호화한다.

$$E_{H(Ki)}(H(RootI)) \quad (10)$$

step3) $H(H(K_i))$을 파일명으로 하고 루트 I블록을 암호화한 최종 R블록을 생성한다.

(5) 저장된 블록 탐색 방법

제안한 인코딩 블록 방법에 따라 파일은 분리되고, 분리된 파일은 다른 피어에게 분산된다. 분산된 블록을 탐색하기 위해서는 다음과 같은 절차를 수행한다.

step1) 이용자는 원하는 파일의 키워드를 입력하고, 입력된 키워드는 160bit RIPE-MD로 해쉬한다. 식 (9)와 같이 두 개의 해쉬값을 생성한다.

step2) 파일명이 $H(H(K_i))$의 값인 데이터를 탐색하여 전송받고, 전송받은 파일을 키 $H(K_i)$로 복호화한다.

step3) 복호화한 결과 루트 I블록을 얻게 되고, I블록이 지시하고 있는 D블록 또는 I블록을 탐색하여 전송 받는다.

step4) D블록을 모두 전송받아 복호화한 다음 블록을 순서대로 연결하여 원본 파일을 생성한다.

3.3. 제안 시스템 인코딩 블록 분산 방법

(1) PLS

PLS(Peer List Server)는 P2P 오버레이 네트워크에 참여하고 있는 활성 피어들에 관한 정보를 담고 있는 초기 접속서버이다. 중앙집중식 P2P의 중앙서버는 전체 네트워크를 구성하고 피어의 접속 목록, 피어의 공유파일 목록, 질의 처리 등에 직접적인 간섭을 하는것에 반면 비중앙집중식 P2P의 PLS는 현재 네트워크에 참여하고 있는 활성 피어의 IP주소 목록만을 유지하여 새로운 피어가 네트워크에 접속을 원할 때 IP주소 목록을 전송하여 네트워크 참여를 돕는다.

(2) 피어 추가 방법

인코딩된 블록을 분산하기 위하여 피어의 위치를 링형의 구조적 모델로 구성한다. 제안 시스템에서는 링형의 구조적 모델을 구성하기 위하여 피어들 마다 고유한 피어 식별자를 생성한다. 다음은 피어 식별자 생성 방법이다.

step1) 제안 시스템에 접속을 원하는 피어는 PLS에 접속을 요청한다.

step2) PLS는 피어의 IP주소 충돌이 있는지 확인하고, 충돌이 없을 경우 피어 식별자와 피어 식별자의 라우팅 테이블을 생성하여 접속을 요청한 피어에게 알려준다.

step3) PLS는 새로운 피어의 피어식별자를 식 (11)에 의하여 검색하고, 해당 피어에게 라우팅 테이블 갱신 메시지를 전송한다.

$$\neq wP_{ID} - \sum_{n=1}^{2^n} \quad (11)$$

- 피어 식별자는 PLS가 랜덤으로 생성한 값을 160bit RIPE-MD로 해쉬한 값이다.
- 라우팅 테이블은 '$P_{ID} + 2^n$'연산을 수행하여 테이블을 생성한다.

(3) 인코딩된 블록 분산 방법

익명성 제공을 위하여 제안 시스템은 인코딩 블록을 다른 피어에게 분산한다. 제안 시스템은 피어 식별자와 블록 식별자를 160bit RIPE-MD 해쉬 값으로 구성하기 때문에 해쉬 값을 기준으로 블록을 분산한다. (그림 3)의 라우팅 테이블의 숫자는 160bit의 해쉬값을 간략하게 표현한 것이다. 다음은 블록 분산에 대한 예이다.

step1) 피어 식별자 '5'는 파일을 출판하여 공유를 요청하였다.

step2) 요청된 파일은 3.2장에서 제안한 인코딩 방법을 수행하였고, 수행 결과 블록 '15'를 생성하였다.

step3) 피어 '5'는 블록 '15'를 저장할 피어를 결정하기 위하여 라우팅 테이블을 참조한다. 참조 결과 식 (12)을 만족하는 피어가 존재하지 않는다. 따라서 피어 식별자가 가장 멀리있는 피어 '10'에 블록을 전송한다.

$$P_{ID} \leqq B_{ID} \qquad (12)$$

step4) 피어 '10'의 라우팅 테이블을 참조한 결과 피어 '17'이 조건에 만족한다. 그러나 2^n의 연산에서 라우팅 테이블이 가르키는 피어는 정확성이 떨어지기 때문에 블록과 피어 식별자가 일치하는 피어를 선택하고, 만약 조건에 만족하지 않을 경우 블록 식별자보다 작은 피어인 '13'을 선택하여 블록을 전송한다.

step5) 피어 '13'은 블록 '15'의 값으로 지정하고 있으며 지정된 피어 '17'에 블록을 전송한다.

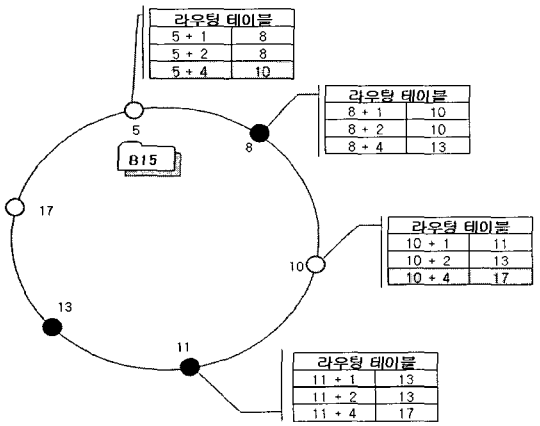
step6) 블록을 전송받은 피어는 인근 두 피어에게 블록을 복사하여 유지시킨다.

(4) 인코딩 블록 탐색 방법

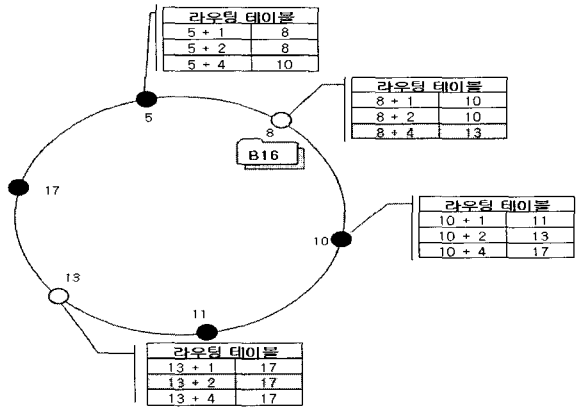
위와 같이 구조적으로 구성된 위상에서 블록과 피어 식별자를 기준으로 블록을 분산하였다. 따라서 분산된 블록을 탐색하기 위하여 블록 분산 방법과 동일한 알고리즘으로 탐색을 수행한다. (그림 4)는 인코딩 블록 탐색을 위한 방법이다.

step1) 제안 시스템 사용자는 블록 탐색을 위하여 키워드를 입력하고, 입력된 키워드는 3.2.5장의 방법에 의하여 해쉬값을 생성한다. 생성된 키워드는 '16'이라고 가정한다.

step2) 피어 '8'은 블록을 라우팅 테이블에서 참조한다. 블록 '16'과 가장 근접한 피어는 '13'이다. 따라서 피어 '13'에 블록 탐색



(그림 3) 인코딩 블록 분산 방법



(그림 4) 인코딩 블록 탐색 방법

메시지를 전송한다.

step3) 피어 '13'은 라우팅 테이블을 참조한다. 그러나 블록 '16'에 대한 정보는 없고, 피어 '17'은 블록보다 크다. 따라서 피어 '13+2'가 지시하는 피어 '17'에 탐색메시지를 보낸다.

step4) 피어 '17'에서 블록은 검색되고, 검색된 블록은 경유한 피어를 역행하여 전송한다.

4. 제안 프로토콜 검토 및 고찰

본 논문에서는 GNUnet 성능 개선을 위하여 운영방법을 기준으로 2가지 관점에서 성능을 비교 분석하였다. 비교 분석의 관점은 첫째, 가변적인 인코딩 방법과 고정크기의 인코딩 방법의 성능을 비교 분석하였다. 둘째, 비구조적 모델에 의한 블록 분산 방법과 구조적 모델에 의한 블록 분산의 차이점을 비교하였고, 제안 시스템의 성능을 평가하였다.

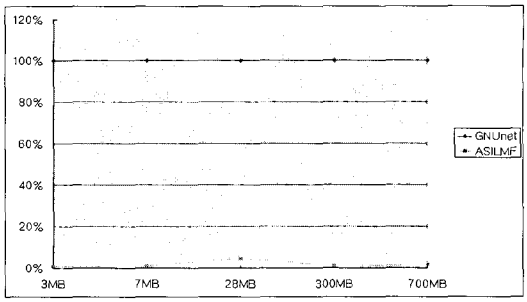
(1) 인코딩 성능 비교 분석

제안 시스템의 성능평가를 위하여 인코딩 과정에서 생성되는 블록의 수를 측정하였다. 먼저 GNUnet의 블록은 1KB 크기로 고정되어 있기 때문에 파일 용량별로 D, I, R블록의 수를 측정하였다. 그리고, 제안 시스템의 블록 수는 가변적인 블록 크기 결정방법을 적용후 결정된 크기에 따라 생성되는 블록의 수를 측정하였다. 인코딩 블록 수를 측정한 결과 (표 2)와 같았다.

(그림 5)는 D블록의 생성 비율을 측정한 결과

(표 2) 파일용량별 블록 수

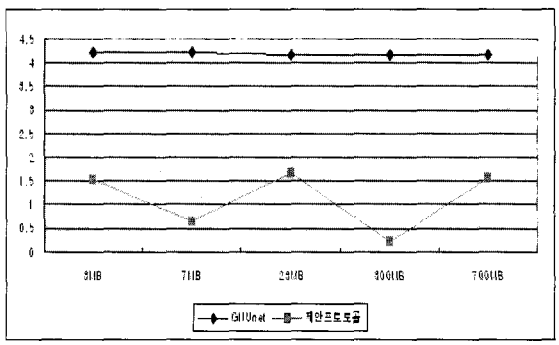
파일용량	GNUnet		제안 프로토콜	
	D블록	I,R블록	D블록	I,R블록
3MB	3072	130	24	2
7MB	7168	301	14	2
28MB	28672	1197	448	20
300MB	307200	12802	600	26
700MB	716800	29869	11200	468



(그림 5) 인코딩 블록 백분율로 표시

값이다. GNUnet에서 생성된 블록 수를 100% 기준으로 측정한 결과 제안 시스템에서 생성되는 블록의 수는 5% 미만이였다. 이와 같이 생성되는 블록의 수 최소화에 의한 성능향상은 생성된 블록을 피어에게 분산하기 위하여 협약되는 피어들의 통신 횟수를 최소화한 것이다. 인코딩된 블록을 분산 및 탐색과정에서 존재하는 블록 수에 따라 분산과정과 탐색과정을 수행해야 하기 때문에 블록 수의 최소화는 네트워크 트래픽을 최소화할 수 있는 방안이다.

(그림 6)과 같이 추가적으로 생성되는 I, R블록의 비율은 기존 GNUnet의 4%에 해당하는 추가 블록 생성을 1% 이내로 최소화 하였다. 추가되는 인코딩 블록의 최소화는 인코딩된 블록의 분산 및 탐색과정에서 맺어야 하는 피어들의 협약 과정을 줄였기 때문에 네트워크에서 발생할 수 있는 협약에 따른 트래픽을 최소화하였다.



(그림 6) 추가 블록 생성(백분율)

(2) 블록 분산 및 탐색 방법 검토 및 고찰

제안 시스템의 블록 분산 및 탐색방법는 두 가지 관점에서 검토 및 고찰이 될 수 있다.

첫째, 구조적 모델에 의하여 피어의 위치를 구성한 제안 시스템은 인코딩된 블록을 분산할 경우 해쉬값을 기준으로 인코딩 블록을 저장시킬 피어를 결정하게 되고, 결정된 피어 정보를 기준으로 인코딩된 블록을 저장하게 된다. 이는 GNUnet의 블록 분산을 위하여 인근 두 피어를 선택하여 인코딩된 블록을 전송하면서 발생하는 2^n의 피어가 블록을 저장하는 문제점을 해결하였다.

둘째, 분산된 블록 탐색을 위하여 GNUnet에서는 블록 저장자의 위치를 추정할 수 있는 방법이 없기 때문에 브로드캐스트방식에 의한 블록 탐색을 진행한다. 하지만 브로드캐스트방식은 인근 피어들에게 모두 탐색 메시지를 전송하기 때문에 과도한 네트워크 트래픽이 발생하는 구조로 이루어졌다. 따라서 제안 시스템의 블록 분산 및 탐색 방법은 네트워크 트래픽 최소화를 위하여 설계하였고, 피어 ID를 기준으로 인코딩된 블록을 분산하였기 때문에 블록 탐색 방법은 인코딩 블록의 해쉬값을 기준으로 인코딩 블록을 저장하고 있는 피어를 직접 접근하여 요구하기 때문에 네트워크 트래픽을 최소화할 수 있었다.

(3) 보안성에 대한 고찰

마지막으로 제안 시스템의 안전성에 대한 고찰이다. 제안 시스템은 파일을 인코딩하여 피어들에게 분산하기 때문에 공격자에 의하여 인코딩된 블록이 도청될 수 있다. 그러나 인코딩된 블록은 익명성 제공을 위하여 내용은 암호화하였고, 파일명은 해쉬 값으로 생성되었다. 따라서 인코딩된 블록이 공격자에게 도청당해도 공격자는 인코딩된 블록을 복호화를 위한 키가 없기 때문에 복호화가 불가능하고, 만약 복호화에 성공하여도 원본파일의 일부 블록이기 때문에 원본파일로 재구성할 수 없다.

5. 결론 및 향후과제

GNUnet은 익명성 제공을 위하여 파일 인코딩 방법과 인코딩된 블록을 다른 피어들에게 분산한 방법을 제안하였다. 그러나 파일을 1KB로 분리하는 방법은 대용량의 멀티미디어 파일 분리에는 적합하지 않다. 700MB의 대용량 멀티미디어의 경우 1KB로 인코딩 할 경우 716,800개의 블록을 생성하고, 추가적인 블록으로 29,869개을 생성한다. 이것은 원본파일의 *4%*에 해당되는 부가적인 저장 공간을 요구하였다. 그리고, 분리된 인코딩 블록을 네트워크에 전송하기 위하여 참고문헌(6)의 개념을 도입했고, 분산된 인코딩 블록을 제공받기 위하여 네트워크망을 혼잡하게 하는 브로드케스트 방식을 사용한다.

따라서 GNUnet의 성능향상을 위하여 제안 시스템은 파일 인코딩 방법과 인코딩 블록을 분산하는 방법을 제안하였다. 인코딩 방법은 "$1KB \leq 2^n \leq 1MB$"의 가변적인 범위의 블록을 결정하는 인코딩 방법을 제안하였고, 결과적으로 추가 블록의 생성을 1% 이내로 최소화 하였다. 그리고, 분리된 블록을 분산하기 위하여 링형의 구조화된 모델을 도입하여 블록 분산 및 탐색에서 발생하는 부하를 최소화하기 위한 방안을 제안하였다.

향후 연구로는 R블록에 의한 키워드 추출은 파일명을 기반으로 생성된다. 따라서 공유된 원본파일의 파일명이 동일하지 않을 경우 인코딩된 블록을 복구하지 못한다는 문제점이 있다. 이 문제점 해결을 위하여 새로운 키워드 추출방법에 대한 연구가 필요하다.

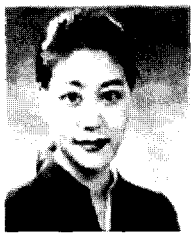
참 고 문 헌

[1] Dennis Kugle, "An Analysis of GNUnet and the Inplications for Anonynous, Censorship-Resistant Networks", PET, 2003.

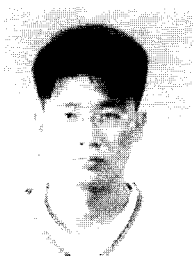
[2] Krista Bennett, Christian Grothoff, Tzvetan Horozov, Ioana Patrascu, "Efficient Sharing

of Encrypted Data", ACISP 2002.
[3] Krista Bennett, Christian Grothoff, "practical anonynous networking", PET,2003.
[4] Christian Grothoff, Krista Grothoff, Tzvetan Horozov, J. T. Lindgren, "An Encoding for Censorship-Resistant Sharing"
[5] Jianning Yang, "APTPFS : Anonynous Peer-to-Peer File Sharing" April, 2005.
[6] 앤디 오람, "Peer-to-Peer 차세대 인터넷 P2P", pp169~177, 2001.
[7] Ion Stoica, Robert norris, David Liben- Nowell, David R.Krishnan, "Chord : a Scalable Peer-to-Peer Lookup Protocol for Internet Applications", IEEE/ACn Tansactions on Networking, Vol. 11, No. 1, February 2003.
[8] David R. Karger, natthias Ruhl, "Dininished Chord: A Protocol for Heterogeneous Subgroup Fornation in Peer-to-Peer Networks", 2003.
[9] A. Keronytis, N.Provos, "The Use of HnAC-RIPEMD-160-96 within ESP and AH", rfc 2857, June, 2000.
[10] Gnutella Developer Forum, "The Gnutella Protocol Specification v0.41," Document. Revision 1.2.

◐ 저 자 소 개 ◑



이 윤 진
1996년 배재대학교 응용수학과 졸업
2000년 배재대학교 컴퓨터공학과 석사
2003년 ~ 배재대학교 컴퓨터공학과 박사과정
관심분야 : 정보보호, 네트워크보안
E-mail : gomyung@pcu.ac.kr



이 명 훈
2001년 배재대학교 컴퓨터공학과 학사
2003년 배재대학교 컴퓨터공학과 석사
2005년 ~ 현재 배재대학교 컴퓨터공학과 박사과정
관심분야 : 정보보호, 컴퓨터네트워크, 전산조직응용
E-mail : lopez@mail.pcu.ac.kr

조 인 준
1982년 : 전남대학교 계산통계학과 공학사
1985년 : 전남대학교 전자계산학과 공학석사
1999년 : 아주대학교 컴퓨터공학과 공학박사
1983년~1994년 한국전자통신연구원 선임연구원
1994년~현재 배재대학교 컴퓨터공학과 교수
관심분야 : 정보보호, 컴퓨터네트워크, 전산조직응용
E-mail : injune@mail.pcu.ac.kr