제안 인코딩 블록을 구조적 P2P 모델에 적용한 GNUnet 성능 개선방안

이명훈*, 박병연**, 조인준*

*배재대학교, **한국과학기술정보연구원

The study on improve method of GNUnet for Structured P2P Model of encording block

Myoung-hoon Lee, Byung-yeon Park, In-June Jo,

*Dept. of Computer Engineering, Paichai Univisity. **KISTI

e-mail : lopez@mail.pcu.ac.kr, bypark@kisti.re.kr, injune@mail.pcu.ac.kr

요 약

GNUnet은 모든 피어에게 익명성(출판자, 저장자, 요구자) 지원방법을 제안한 대표적인 P2P 시스템이다. GNUnet은 익명성 제공을 위하여 파일을 1 Kbyte 블록 단위로 분리하여 인코딩하였고, 인코딩된 정보를 주변 피어에게 분산시키는 방법을 채택하였다. 그러나, 대용량화 되는 멀티미디어 파일을 1 Kbyte의 크기로 분리하는 것은 추가적인 블록의 생성을 의미한다. GNUnet에서 제안하고 있는 파일 분리 방법을 분석한 결과 피어들의 검색을 위하여 제공하는 R블록, I블록등이 원본 파일의 4%에 해당되는 저장공간을 추가적으로 요구하고, 분리된 블록을 전송하기 위하여 추가적인 네트워크 트래픽이 발생하였다.

본 논문에서는 이러한 문제점 해결을 위한 새로운 인코딩과 라우팅 방안을 제안하였다. 제안 인코딩 방법은 블록을 생성할 때 발생하는 추가적인 블록을 기존의 4%에서 1%이내으로 최소화 하였고, 분리된 블록을 분산할 때 발생하는 네트워크 트래픽 최소화를 위한 구조적 위상 방법을 채택하였다. 결론적으로 저장공간의 낭비 및 네트워크 트래픽을 최소화할 수 있다.

ABSTRACT

The GNUnet in P2P system have solved an anonymous(publisher, storer, demander); a service of condition equal for peer. The GNUnet for an anonymous has separated a file and dispersed to the network. But, the 1Kbyte block size of the GNUnet is a creation of many additioinal block. I and R block has created with indirect point of D block The waste of I and R block appeared to 4% of the original file and the additional network traffic for the block transmission.

To resolve the problems, this paper proposes an new scheme of file splitting distribution using P2P networks with the new GNUnet protocol, The GNUnet support minimization of the additional block and for a netwodrk traffic. It proposed an efficiency improvement of encording block and routing algorithm.

키워드

P2P, 익명성, GNUnet, Chord, 인코딩, 위상

I. 서론

P2P 서비스는 응용 소프트웨어를 통해 PC내의 파일을 서로 공유하는 시스템을 의미한다. 초기 P2P 서비스인 냅스터와 소리바다는 서버를 도입하여 각 피어가 자신의 파일 정보를 중앙서버에 저장하는 방법으로 서비스를 제공하였다. 연구결과 이 방법은 30%에 해당하는 특정피어가 모든 정보를 제공하였고 동등한 사용자 관점의 P2P 시스템의 원칙을 위반하게 되었다. 따라서 모든 사

용자가 동일한 저장공간과 자유로운 정보공유에 대한 연구가 진행되었고, 대표적인 서비스로 Freenet과 GNUnet을 들수있다. GNUnet은 자유로운 정보공유를 서비스하기 위하여 익명성[3][5] 지원방법을 제안하였다. 익명성이란 파일을 공유하는 출판자와 파일을 저장하는 저장자, 파일을 요구하는 요구자에 대한 비밀을 보장하는 것이다.

출판자의 익명성은 파일을 공유하는 피어가 파일을 출판자가 저장하는 것이 아니고, 파일을 인코딩하여 다른 피어에게 저장을 요청하는 방법으로써 해결하였다. 저장자의 익명성은 인코딩 블록을 해쉬와 암호화방식을 적용했기 때문에 저장자는 인코딩 블록의 내용을 확인할 수 없다. 따라서 인코딩 블록 저장자는 블록에 대한 법적 소유권을 거부할 수 있다. 요구자의 익명성은 일반적인 키워드를 해쉬하여 인코딩된 블록을 검색하기 때문에 공격자가 해쉬 값을 도청할 경우 요청자가 원하는 키워드가 노출되지 않는다. 따라서 요청자의 익명성을 제공한다.

결론적으로 GNUnet은 익명성 제공을 위하여 파일 인코딩 방법과 인코딩된 블록의 전송방법을 제안하였다. 그러나 GNUnet에서 제안하고 있는 인코딩 방법은 대용량의 멀티미디어 파일에는 비합리적이고, 인코딩 블록을 비구조적인 방식으로 분산하기 때문에 과도한 네트워크 트래픽의 원인이 되었다.

본 논문은 GNUnet의 불필요한 저장공간 및 과도한 네트워크 트래픽을 해결하기 위하여 인코딩 블록 크기를 "210 ≤ 2n ≤ 220" 범위에서 결정하는 방법을 제안하였고, 블록 분산방법으로 Chord[7]에서 제안한 링형태의 구조적 모델을 새롭게 적용하였다. 이를 위하여 II장에서는 GNUnet의 인코딩 방법 및 파일 분산방법 그리고 구조적 위상을 제안한 Chord에 대하여 설명하였다. III장에서는 이 논문에서 제시하는 성능개선 방법을 설명한다. IV장에서는 제안 프로토콜을 검토 및 고찰하였고, 마지막으로 결론을 맺었다.

II. 관련 연구

2.1 GNUnet 파일 인코딩 방법[1][2]

GNUnet[1][2]은 익명성 지원을 위하여 파일을 1Kbyte의 고정길이 블록으로 분리하고, 분리된 블록을 분산하는 방법을 제안하였다. 제안된 인코딩 방법은 D(data)블록, I(Indirection)블록, R(root)블록의 3가지 블록을 생성하고, 생성된 블록은 네트워크의 다른 피어에게 전송된다. D블록은 다른 피어에게 분산되기 때문에 D블록을 지시하기 위한 I블록을 추가적으로 생성되어야 하고, 이용자 검색의 편리성 제공을 위하여 R블록을 제공한다. 그러나, 원본 파일의 추가 블록인 I, R은 4%에 해당되는 추가적인 저장공간을 요구한다.

2.2 GNUnet 인코딩 파일분산 및 탐색방법 [4]

인코딩된 블록은 출판자의 익명성 제공을 위하여 다른 피어에게 블록을 저장한다. GNUnet에서 제안한 파일 분산 방법은 피어의 라우팅 테이블을 기준으로 랜덤하게 2개의 피어를 선택하여 블록을 전송하고, 참고문헌[6]의 방식을 적용하여 TTL 값을 생성한다. GNUnet은 TTL값이 커질수록 저장되는 피어의 수가 2n으로 증가된다. 이러한 비 구조적방법의 블록 분배방법은 블록 탐색에 있어 특정한 규칙이 없기 때문에 광고방법에 의하여 블록을 탐색하여야 하고, 네트워크의 트래픽을 증가시킨다.

3. Chord 파일 정보 분산 및 탐색 방법[7]

Chord 시스템은 P2P 응용을 위한 분산처리 자원 탐색 프로토콜이다. Chord 시스템은 피어의 위치를 링형의 구조적 모델로 구성하고, 피어-ID를 중심으로 파일정보를 제공한다. 이러한 서비스를 제공하기 위하여 피어-ID와 파일정보는 SHA-1 알고리즘과 같은 해쉬함수를 이용하여 해쉬되고, 파일정보의 해쉬 값을 피어-ID와 비교하여 근접한 피어를 결정한다(파일정보 ≤ 피어-ID). 따라서 저장된 파일정보를 탐색하기 위해서는 파일정보의 해쉬값과 근접한 피어-ID를 탐색하여 파일정보를 제공받는다.

III. 제안 시스템 구조

GNUnet은 파일의 인코딩 과정에서 1Kbyte라는 단위로 블록을 분리한다. 이 블록은 저용량의 파일의 경우 문제가 없지만 대용량 멀티미디어 파일의 경우 I블록과 R블록 생성에 따른 저장 공간의 낭비가 발생한다. 동영상 멀티미디어 파일의 기본 크기인 700Mbyte를 예로 들어 보겠다. 만약 700Mbyte 파일 저장을 요구할 경우 716,800개의 D블록으로 파일을 분리한다. 그리고, D블록을 간접적으로 지시하는 I블록은 29868개가 생성된다. 이와 같이 분리된 블록은 비구조적 위상에서 2개의 이웃 피어를 선택하여 블록을 전송하고, TTL 값에 의하여 2n개의 피어에게 전송한다. 따라서 I, R블록 및 비구조적 방법의 파일 분리 방법에 따른 추가된 저장공간이 요구되었고, 비구조적 방법으로 분산된 블록의 탐색에서 광고방법을 사용하기 때문에 과도한 네트워크 트래픽이 발생하였다.

본 논문에서는 저장 공간의 낭비와 트래픽 발생을 줄이기 위한 방법으로 새로운 인코딩 방법과 인코딩된 블록을 구조적 위상으로 구성된 피어에게 분산하는 방법을 제안하였다.

1. 제안 시스템 인코딩 방법

제안 인코딩 방법은 추가적으로 생성되는 I, R 블록의 추가되는 저장공간을 최소화하기 위하여 블록의 크기를 기존 1Kbyte에서 "$1Kbyte \leq 2^m \leq 1Mbyte(2^{10} \leq 2^m \leq 2^{20})$"의 크기로 제안하였다. 네트워크에서 전송되는 파일의 크기는 1Kbyte를 기준으로 전송되기 때문에 단편화 최적화를 위한 2m으로 결정하였고, 최대 1Mbyte의 크기는 사용자가 공유하는 저장 공간의 크기를 고려하여 결정하였다.

[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, m	대입되는 순차 값
P(XX)	피어를 식별하기 위한 피어-ID
B(XX)	파일을 해쉬한 블록의 해쉬 값

3.1.1 인코딩 블록 크기 결정

GNUnet의 인코딩 방법은 1Kbyte의 고정길이로 인코딩 블록 생성 방법을 제안하였다. 그러나 1Kbyte의 블록은 4%의 추가 블록을 생성하였고, 추가적인 저장공간을 요구하였다. 따라서 제안 시스템에서는 인코딩 블록의 크기를 1Kbyte $\leq 2^m \leq$ 1Mbyte의 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 25^m으로 나누고, 나눈 값이 "1Kbyte $\leq$ BX $\leq$ 1Mbyte" 의 범위에 결과 값을 구한다.

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

step2) 제안 시스템은 단편화 최적화를 위하여 2^m의 크기로 구성된다. step1에서 결정된 블록의 크기는 2^m의 결과 값이 아니기 때문에 식 (2)의 조건을 만족하는 BC의 값을 구한다. BC의 범위는 1Kbyte $\leq$ BC $\leq$ 1Mbyte이다.

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

step3) 결정된 인코딩 블록의 크기 BC에 따라 파일을 분리한다.

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

3.1.2 D블록 생성

결정된 BC의 크기에 따라 분리된 블록은 저장자의 익명성 보호를 위하여 블록의 데이터는 암호화하고, 파일명은 160비트 RIPE-MD로 해쉬한다. 저장자의 익명성 제공을 위하여 블록의 데이터는 암호화하고, 파일명은 160bit로 해쉬한 값이다. 다음은 D블록 생성 방법이다.

step1) 파일이 분리된 각 블록 B_i를 160비트 RIPE-MD[8]로 해쉬한다.

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

step2) 해쉬 값을 키로 하여 분리된 블록의 데이터를 암호화한다.

$$E_{H(D1)}(B_1),\ E_{H(D2)}(B_2) \cdots E_{H(Dn)}(B_n) \quad (5)$$

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

$$H(E_{H(D1)}(D_1)),\ \cdots H(E_{H(Dr)}(Dn)) \quad (6)$$

3.1.3 I블록 생성

생성된 D블록은 다른 피어에게 분산되어 저장된다. 따라서 분산된 블록을 지시하기 위한 식별자가 필요하다. I블록은 분산된 D블록을 지시하는 블록이다. 다음은 I블록 생성 방법이다.

step1) 각 D블록에 대하여 식 (4)의 평문 해쉬 값과 식 (6)의 암호화된 해쉬값을 저장한다. 총 25개의 D블록의 정보하고, 이 정보는 D블록의 파일명과 암호키 값을 의미한다. 다음은 생성된 I블록의 예이다.

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

step2) CRC32 값을 계산한다. (4Byte)

step3) 생성하고자 하는 I블록과 연결된 모든 해쉬 값을 160비트로 해쉬한 20바이트의 슈퍼해쉬 값을 계산한다.

step4) D블록 또는 I블록을 지시하는 I블록을 생성한다.

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

step5) I블록을 160bit RIPE-MD로 해쉬하여 파일명을 생성한다.

3.1.4 R블록 생성

D블록과 I블록은 파일을 저장하기 위하여 생성된 블록이다. 따라서 사용자 관점에서는 키워드에 의한 검색을 요구한다. R블록은 사용자의 키워드 검색을 제공하기 위하여 구성된 블록이다.

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

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

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

$E_{H(Ki)}(H(R))$ (10)

step3) H(H(Ki))을 파일명으로 하고 루트 I블록의 파일명을 암호화한 최종 R블록을 생성한다.

3.1.5 저장된 블록 탐색 방법

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

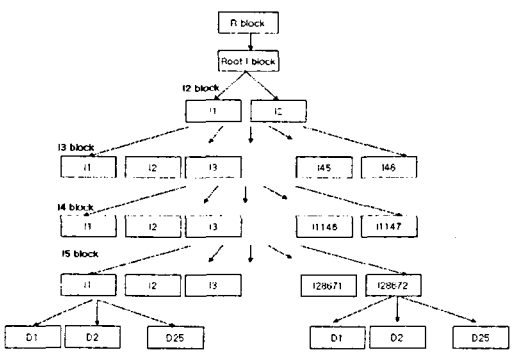
step1) 이용자는 원하는 파일의 키워드를 입력하고, 160bit RIPE-MD로 해쉬한다. 해쉬 값은 식 (9)와 같다.

step2) 파일명이 H(H(Ki))의 값이 데이터를 탐색하여 전송받고, 전송받은 데이터를 H(Ki)를 키로하여 복호화한다.

step3) 복호화한 결과 루트 I블록은 I블록 또는 D블록의 키와 블록의 해쉬값을 지시하고 있고, 지시하고 있는 해쉬 값을 탐색하여 전송받는다. 전송받은 블록은 키에 의하여 복호화하고, 이 과정을 D블록을 모두 받을때까지 진행한다.

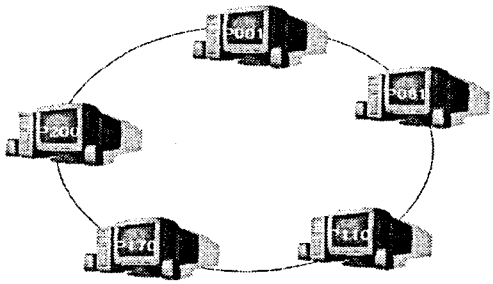
step4) D블록을 모두 전송받아 복호화한 다음 블록을로 순서대로 연결하여 최종 파일을 생성한다.

제안 인코딩 방법에 의하여 블리된 블록의 구조는 (그림 1)의 트리구조로 구성된다.



[그림 1] 제안 시스템 인코딩된 블록

3.2 제안 프로토콜 인코딩 블록 분산 방법

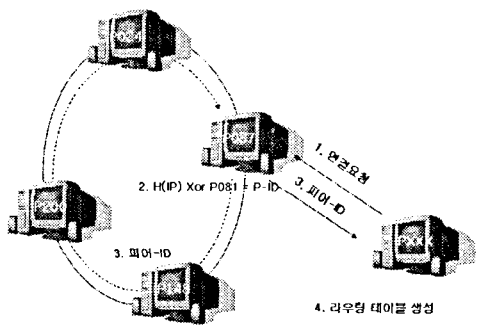


[그림 2] 제안 프로토콜 위상

출판자의 익명성 제공을 위하여 인코딩된 블록은 다른 피어에게 저장한다. 인코딩된 블록은 구조적으로 분산된 피어에게 분산이 되며 피어의 위상은 (그림 2)와 같은 링형을 유지한다.

3.2.1 피어 추가 방법

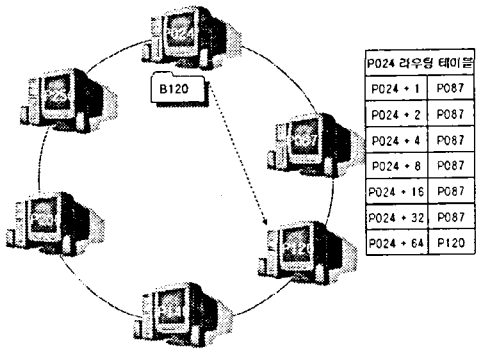
제안 프로토콜에 피어가 추가되었을 경우 추가된 피어는 식별자 피어-ID를 생성한다. 피어-ID는 IP주소를 160bit로 해쉬하고, 해쉬 값에 연결된 피어의 ID 값을 XOR 연산을 수행하여 결정한다.



[그림 3] 피어-ID 생성과정

XOR 연산을 수행하는 이유는 IP주소의 해쉬값은 고유한 해쉬 값을 얻게된다. 하지만, 공격자가 역으로 IP주소를 해쉬하여 피어-ID와 비교할 경우 IP주소가 노출되기 때문에 연결된 피어-ID와 XOR연산을 수행하는 것이다.

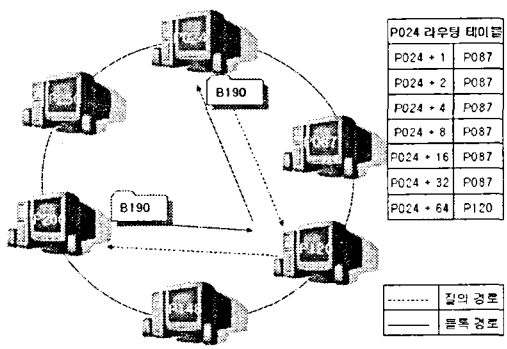
(그림 3)은 피어-ID 생성과정이다. 추가되는 피어는 GNUnet에서 서비스 중인 피어에 연결을 요청하고, 요청을 받은 피어는 피어-ID를 요청자와 GNUnet의 피어들에게 피어정보를 전송한다. 새로운 피어정보를 받은 피어는 피어정보를 갱신하고, 새로운 라우팅 테이블을 생성한다. 라우팅 테이블은 피어-ID에 2^n의 덧셈 연산을 수행한다.



[그림 4] 인코딩 블록 분산 방법

3.2.2 인코딩 블록 분산 방법

출판자의 익명성 제공을 위하여 인코딩 블록은 다른 피어에게 전송해야 한다. 피어-ID는 160bit의 해쉬 값으로 구성되어 있으며, 블록의 해쉬 값과 피어-ID를 비교하여 같거나 작을 경우에 블록저장 피어로 결정한다.



[그림 5] 블록 검색 방법

(그림 4)는 인코딩 블록 분산에 대한 예이다. 6개의 피어로 구성된 링형의 위상에서 P024는 인코딩 블록 B120의 저장을 요청한다. 출판자는 자신의 라우팅 테이블을 조사하여 B120보다 같거나 큰 피어-ID를 검색하고, B120와 가장 가까운 피어-ID에 블록을 전송한다. (그림 4)에서는 P120과 B120이 동일한 피어가 존재하기 때문에 P120에 블록 저장한다.

3.2.3 인코딩 블록 검색 방법

제안 프로토콜에서 블록의 저장은 출판자의 익명성 보호를 위하여 구조적 위상으로 구성된 피어와 블록의 해쉬 값에 의하여 전송되었다. 따라서 분산된 블록을 전송 받기 위해서는 해쉬 값에 의하여 피어를 검색하고, 블록을 전송받는다.

(그림 5)와 같이 피어 P024가 블록 B190을 요청했다. 초기 요청한 피어는 라우팅 테이블을 검토하여 B190과 같거나 큰 피어-ID를 검색한다. 그러나 P024의 라우팅 테이블에는 조건에 만족하는 테이블이 없다. 따라서 라우팅 테이블에서 마지막 피어로 검색 요청을 보낸다. 라우팅 테이블 마지막에 있는 피어는 P120이기 때문에 P120 피어에 B190의 검색을 요청하게 되고, P120은 라우팅 테이블을 검색한다. P120의 테이블을 검색한 결과 P148과 P200의 피어를 가르키고 있다. 여기서 검색요청 메시지는 P148과 P200에 보내게 된다. P200은 B190보다 큰 해쉬 값을 갖기 때문에 조건에 만족한다. 그러나 라우팅 테이블 설계상 2^n의 수치는 중간 피어를 식별할 수 없을 가능성이 존재한다. 따라서 검색의 정확성을 제공하기 위하여 P148과 P200의 피어에 검색 요청을 수행한다.

Ⅳ. 제안 프로토콜 검토 및 고찰

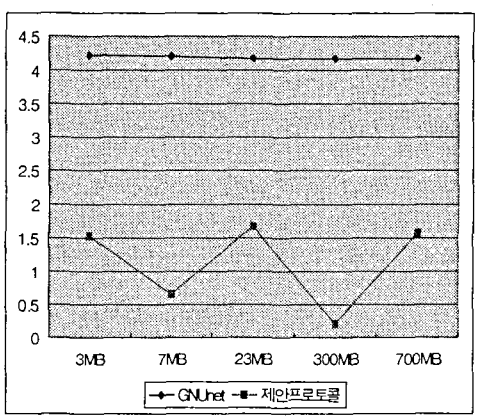
제안한 알고리즘의 타당성을 검토하기 위하여 멀티미디어 데이터의 용량별 인코딩 블록 수와 분리된 블록의 분산과 검색에 발생되는 트래픽량에 대하여 검토 및 고찰하였다. 제안 프로토콜은 하나의 파일을 여러 블록으로 분리하여 피어들에게 분산시키고, 다시 분산된 블록을 하나의 파일로 처리되어야한다. 따라서 효율적인 블록 분산방법이 필요하다. 이를 위하여 저장공간의 낭비를 줄일수 있는 최적의 인코딩 방법과 효율적으로 블록을 분산하고, 분산된 블록을 하나의 파일로 처리하기 위하여 링형의 구조적 위상을 제안하였다.

제안 프로토콜은 2가지 응용분야 문제점을 해결에 대한 검토 및 고찰이다. 즉 I, R블록에 의한 부
가적인 저장공간의 낭비의 최소화와 인코딩된 블록의 저장 및 검색을 위한 향상된 위상에 대한 검토 및 고찰이다. 따라서 이러한 2가지 측면에서 검토 고찰은 다음과 같다.

첫째, 파일 크기의 대형화에 따라 블록을 지시하는 I, R블록의 저장공간 낭비이다. (그림 6)은 멀티미디어 데이터에 추가되는 블록에 대한 내용이다. 현재 GNUnet에서 제안된 인코딩 방식을 적용했을 경우 "4%"의 저장공간 낭비가 발생하였다. 그러나 제안 프로토콜을 적용할 경우 "1%" 이내인것을 확인할 수 있다. 따라서 제안 프로토콜을 GNUnet에 적용할 경우 간접 지시자 블록을 저장공간 낭비를 줄일 수 있다.

[표 2] 파일용량별 블록 수

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



[그림 6] 간접 지시자 생성 비율

둘째, 분리된 블록에 대하여 저장 및 검색 기능을 제공하기 위하여 효율적인 위상을 제안하였다. 기존 위상의 방식은 근접 라우팅 테이블에서 랜덤하게 피어를 선정하여 블록을 전송하기 때문에 분산된 블록을 하나의 파일로 처리하기 위해서는 브로드캐스트 방법으로 인코딩 블록을 요청해야 된다. 따라서 브로드캐스트를 피어가 수행했을 경우 검색 블록수가 많을 경우 GNUnet 시스템은 매우 혼잡하게 되었고 피어들은 자신의 시스템이 느려지는 현상을 체험하게 되었다. 이러한 혼잡현상을 최소화하기 위하여 제안 프로토콜에서는 링형의 구조적 위상을 제안하였다. 링형의 구조적 위상은 블록의 저장 및 검색에서 특정피어의 피어-ID와 검색하고자 하는 블록-ID를 비교하여 블록-ID보다 같거나 큰 피어-ID를 만났을때 저장 및 검색을 수행한다. 따라서 기존의 브로드캐스트 방식이 아닌 특정피어를 지정하는 방식이다. 이 방식을 적용했을 경우 다수의 피어가 검색을 수행해도 특정피어에 대하여만 트래픽이 발생하고, 요청을 받지 않은 피어들은 서비스 대기상태에 놓인다. 제안 위상방법을 GNUnet 시스템에 제공할 경우 네트워크의 혼잡사태를 최소화 할수 있다.

결론적으로 제안 프로토콜은 파일을 블록단위로 분리하여 논리적인 네트워크에 분산과 검색을 수행할 발생하는 저장공간 낭비 및 네트워크 트래픽의 최소화방안을 제시하였다. 따라서 GNUnet에 적용할 경우 저장공간 최소화 및 네트워크 트래픽을 최소화할 수 있다. 그러나 I,R 블록 도청에 위험성과 구조적 위상의 성능 및 안정성에 대한 정확한 방안이 제시되어 있지 않기 때문에 구체적인 구조적 위상의 연구개발이 진행되어야 한다.

V. 결론 및 향후과제

대표저인 P2P 시스템인 Gnutella와 소리바다는 30%에 해당하는 사용자가 전체 파일을 공유하게 되었다. 이것은 P2P의 자원 공유라는 관점에서 위반이 되었고, 모든 이용자의 동등한 자원분배를 위한 익명성 연구가 진행되고 있으며 대표적인 시스템이 GNUnet이다.

GNUnet은 세가지 요구사항을 만족시키기 위하여 제안되었다. 첫째, 출판자의 익명성 보장, 둘째, 컨텐츠 수신자의 익명성 보장, 셋째로, 컨텐츠 저장자가 저장되어 있는 정보에 대한 부인권을 제안하였다. GNUnet은 요구사항을 만족시키기 위하여 II장에서 제시한 파일 분리방법과 분리된 파일을 피어들에게 분산한 방법을 제안하였다. 그러나 파일을 1Kbyte로 분리하는 방법은 대용량의 멀티미디어 파일 분리에는 적합하지 않다. 기본적으로 700Mbyte의 용량을 갖는 동양상 파일을 1Kbyte로 인코딩 할 경우 716800개의 블록을 생성하고, 이를 지시하기 위한 추가적인 블록으로 29869개를 생성한다. 이것은 원본파일의 4%에 해당되는 부가적인 저장 공간을 요구한다. 그리고, 분리된 인코딩 블록을 네트워크에 전송하기 위하여 믹스마스터[6]의 개념을 도입했고, 분산된 인코딩 블록을 제공받기 위해서는 브로드케스트 방식을 사용한다. 이 방식은 네트워크 망을 매우 혼잡하게 한다. 따라서 GNUnet은 멀티미디어 파일에는 적합하지 않았다.

본 논문은 GNUnet의 파일 인코딩 방법과 인코딩 블록을 분산하는 방법을 제안하였다. 인코딩 방법은 $2^{10} \leq 2^{n} \leq 2^{20}$의 범위의 블록크기를 결정하여 인코딩 하였다. 기존에 인코딩 블록을 지시하기 위한 부가적인 공간으로 4%를 소비했지만, 제안 프로토콜을 도입할 경우 1% 이내로 저장공간 낭비를 최소화 하였다. 그리고, 분리된 파일의 분산방법으로써 Chord에서 제시한 구조화된 모델을 도입하였다. 피어들이 GNUnet 시스템에 참여할 경우 피어 ID를 부여받게 되고, 이를 근거로 링형태의 구조에 참여하게 된다. 참여한 피어는 '피어ID'를 근거로 '파일ID'가 같거나 작은 파일을 저장하게 된다. 따라서, 네트워크에 분산되어 있는 피어들은 링형태를 구성하기 때문에 파일의 저장및 검색 능력이 기존 비 구조적방법과 비교할 경우 빠른 검색과 네트워크 트래픽 최소화의 장점을 갖고있다. 결과적으로 제안하는 시

스템은 GNUnet의 성능향상을 위하여 저장공간과 네트워크 트래픽을 최소화하는 방안을 제안하였다.

본 논문의 향후과제로써 제안한 인코딩 방법은 D블록의 크기와 I,R 블록의 크기다 다르기 때문에 공격자는 R블록을 도청하여 요청자가 원하는 블록을 제공받을수 있다. 이러한 문제를 해결하기 위하여 R블록의 보안성을 강화해야 한다. 그리고 또 다른 과제로써 강제 종료되는 피어를 들수 있다. 피어가 갑자기 서비스를 종료하면 그동안 저장되어 있던 블록을 검색할 수가 없다. 따라서 강제 종료되는 피어에 대한 문제를 해결하기 위한 연구가 필요하다.

참고문헌

[1] Dennis Kugle, " An Analysis of GNUnet and the Implications for Anonymous, 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 anonymous networking", PET,2003.

[4] Christian Grothoff, Krista Grothoff, Tzvetan Horozov, J. T. Lindgren, "An Encoding for Censorship-Resistant Sharing"

[5] Jianning Yang, "APTPFS : Anonymous Peer-to-Peer File Sharing" April, 2005.

[6] 앤디 오람, "Peer-to-Peer 차세대 인터넷 P2P", pp169~177, 2001.

[7] David R. Karger, Matthias Ruhl, "Diminished Chord: A Protocol for Heterogeneous Subgroup Formation in Peer-to-Peer Networks", 2003.

[8] A. Keromytis, N.Provos, "The Use of HMAC-RIPEMD-160-96 within ESP and AH", rfc 2857, June, 2000.