Unchoked Peer 개수에 따른 BitTorrent 성능 분석

정회원 정 태 중*, 한 진 영*, 김 현 철*, 권 태 경*, 종신회원 최 양 희*

An Analysis on BitTorrent Performance Based on the Number of Unchoked Peers

Taejoong Chung*, Jinyoung Han*, Hyunchul Kim*, Taekyoung "Ted" Kwon* ***Regular Members***, Yanghee Choi* ***Lifelong Member***

요 약

최근 파일 공유를 위해 널리 사용되는 BitTorrent의 강점은 BitTorrent의 핵심 메커니즘인 peer 선택 기법에 기인한다. Peer 선택 기법은 해당 호스트가 업로드 할 peer를 선택하는 것인데, 이 때 자신에게 각 peer가 업로드한 비율을 비교해서 가장 많이 업로드한 peer들에게 업로드를 하게 된다. 그런데 BitTorrent에서는 업로드할 peer의 개수 (즉, unchoked peer 개수)를 4개로 고정하고 있다. 하지만 peer들의 개수나 다운로드 속도, 업로드 속도와 같은 공유집단 (즉, swarm)의 특성은 계속 변하기 때문에 unchoked peer 개수를 고정하는 것은 효율적이지 않다. 본 논문에서는 BitTorrent에서 unchoked peer 개수가 고정된 4개가 아니라 변할 때 전체 시스템 성능에 어떠한 영향이 끼치는지 분석한다. 분석을 위해 본 논문에서는 서울대학교 캠퍼스 망에 테스트베드를 설치하였고, 실제로 널리 사용되는 오픈 소스 BitTorrent 클라이언트를 수정해서 사용하였다. 테스트베드에서의 다양한 실험을 통해 BitTorrent의 성능이 unchoke 개수에 따라 달라지게 되는 것을 보였고, 이에 따라 swarm의 상황을 고려해서 unchoke개수를 적절하게 조절하는 메커니즘이 필요하다는 것을 보였다.

Key Words : BitTorrent, Swarming System, Choking Algorithm, Number of Unchoked Peer

ABSTRACT

A strength of BitTorrent, which is widely used for file sharing today, is due to its peer selection mechanism which is designed to encourage peers to contribute data. In the peer selection phase in BitTorrent, peers to upload the file in a swarm are selected by determining which peers upload the most to themselves. However, the number of peers to upload (i.e., number of unchoked peers) is fixed to four in its peer selection mechanism of BitTorrent, which yields inefficiency because the situation of the swarm may vary frequently (e.g., number of peers in the swarm, download rates, and upload rates). In this paper, we analyze the swarming system performance when the number of unchoked in BitTorrents is not static, but dynamic. For empirical investigation, we established a testbed in Seoul National Universty by modifying an open-source BitTorrent client, which is popular. Through our experiments, we show that an adaptive mechanism to adjust the number of unchoked peers considering the situation of the swarm is needed to improve the performance of BitTorrent.

※ This work was supported by NAP of Korea Research Council of Fundamental Science and Technology, and the project K-10-L05-C03-S05 of Korea Institute of Science and Technology Information(KISTI).

* 서울대학교 멀티미디어 및 이동통신연구실(tijayoo@gmail.com)

논문번호 : KICS2010-06-264, 접수일자 : 2010년 6월 11일, 최종논문접수일자 : 2010년 8월 5일

Ⅰ. 서 론

최근 BitTorrent[1]는 파일을 공유하기 위한 peer-to-peer (이하 P2P) 소프트웨어로 많이 사용되고 있다. IPoque에서 2009년에 발간한 보고서에 따르면 현재 인터넷 트래픽의 약 27-55%를 BitTorrent가 차지하고 있다[2]. 이렇게 BitTorrent가 널리 사용되는 이유는 BitTorrent가 기존의 다른 P2P에서 해결하지 못한 문제점들을 해결하였기 때문이다. 첫째, BitTorrent에서는 인기도가 높은 파일에 대한 요구가 몰리는 flash crowd 상황에서도 효과적으로 파일이 전송된다. 둘째, 기존의 P2P에서는 파일을 공유하는 공유집단 (이하 swarm)에 기여하지 않고 받기만 하는 free-rider 문제[3]가 심각한데, BitTorrent에서는 이 문제가 효과적으로 방지된다[4]-[6].

이러한 BitTorrent의 장점들은 choking algorithm이라고도 불리우는 BitTorrent의 핵심 메커니즘인 peer[1]선택 기법에 기인한다. 본 기법은 각각의 peer에게 인센티브를 제공함으로써 파일을 공유하도록 촉진 하는 built-in 인센티브 메커니즘으로, 본 기법의 효율성은 많은 연구에서 증명되었다[4]-[7].

그런데 BitTorrent에서 기본적으로 호스트가 여러 peer로부터 다운로드를 할 수 있지만, 자신이 업로드 할 peer의 개수 (즉, unchoked peer[2] 개수)는 4개로 고정하고 있다. 하지만 BitTorrent의 공유집단 (즉, swarm)의 상황은 계속 변하기 때문에 unchoked peer 개수를 4개로 고정하는 것은 효율적이지 않다. 예를 들어서 swarm에 참여하는 peer의 개수나 데이터의 완전한 사본을 가지고 업로드만 해 주는 peer (이하 seed)의 개수, 데이터를 다운로드하고자 하는 peer (이하 leecher)의 개수, 그리고 swarm내에서 다운로드 및 업로드 하는 속도 등은 계속해서 변하게 된다. 이때, 해당 호스트가 다운로드를 받기 위해서는 (즉, 다른 peer가 해당 호스트에게 업로드를 해주기 위해서는) 다른 peer로부터 tit-for-tat 전략에 의해 선택을 받아야 한다. 따라서 다른 peer로 부터 선택을 효과적으로 받기 위해서는 해당 호스트가 swarm의 상황에 따라 소수의 peer에게 많은 양의 대역폭으로 업로드를 하는 것이 유리할 수도 있고, 많은 peer에게 적은 양의 대역폭으로 업로드를 하는 것이 유리할 수도 있다. 하지만 현재의 BitTorrent의 peer 선택 기법은 unchoked peer 개수를 고정 해서 이러한 swarm의 상황을 반영하지 못하고 있다.

본 논문에서는, BitTorrent에서 unchoked peer 개수가 고정된 4개가 아니라 변할 때, 전체 시스템 성능에 어떠한 영향이 끼치는지 분석한다. 분석을 위해 본 논문에서는 서울대학교 캠퍼스 망에 테스트베드를 설치하였고, 실제로 널리 사용되는 오픈 소스 BitTorrent 클라이언트를 수정해서 실험하였다. 실험 결과를 바탕으로 unchoked peer의 개수를 swarm의 상황에 따라 적응적으로 결정하는 기법의 필요성을 제시 하였다.

본 논문의 구성은 다음과 같다. 2장에서는 BitTorrent의 동작과정을 설명하고, 3장에서는 실험을 위한 테스트베드 및 실험 설정에 대해 설명한다. 4장에서는 다양한 시나리오에서 unchoke할 peer 개수 변화에 따른 성능 측정을 하고 마지막으로 5장에서 결론을 맺는다.

Ⅱ. BitTorrent 배경 지식

BitTorrent가 실제로 널리 사용됨에 따라 BitTorrent의 throughput, fairness 그리고 incentive 문제에 관해 많은 연구가 진행되었다[5]-[8]. 본 장에서는 BitTorrent 프로토콜에 관한 용어설명과 더불어 핵심 알고리즘인 peer 선택 기법을 설명하도록 한다.

2.1 BitTorrent 용어 정리

BitTorrent에 관한 많은 연구가 진행되었으나 용어들은 표준화 되어있지 않다. 따라서 명확성을 위해서 본 논문에서 사용할 대표적인 BitTorrent 용어들을 정리하였다.

2.1.1 Torrent / Swarm

BitTorrent 프로토콜을 사용하여 같은 파일을 받고자 하는 peer들의 집합. 보통 torrent와 swarm이 교환적으로 사용된다.

2.1.2 Metainfo File (.torrent)

Torrent file (즉, ".torrent")이라고도 불리우며 piece의 개수나 파일을 download하기 위한 모든 정보가 들어있다. 각각의 piece를 검증하기 위해 piece의 SHA-1 hash값이 들어있으며, tracker의 IP 주소와 port number가 들어있다.

1) 본 논문에서는 peer는 파일을 함께 공유하는 호스트중에서 자신과 연결이 되어 있는 호스트를 지칭한다.

2) 호스트가 peer에게 전송하지 않을 때 해당 peer는 choke 되었다라고 하며, 반대로 호스트가 peer에게 전송하고자 할 때 해당 peer는 unchoke 되었다고 한다.

2.1.3 Tracker

BitTorrent System중에서 유일한 centralized 요소로 실제적인 파일의 전송에는 관여하지 않으나 파일을 공유하는 peer들의 위치 정보를 가지고 있으며, 그들의 통계 또한 수집한다. 호스트가 접속하면 해당 토런트를 공유하는 peer 리스트를 제공한다.

2.1.4 Piece and Blocks

파일은 여러 개의 piece로 나뉘어져 있으며 각각의 piece는 block으로 또 나누어져 있다. 실제적인 전송단위는 block이며 peer가 완전한 piece을 소유하고 있지 않으면 공유할 수가 없다.

2.1.5 Leecher and Seed

파일의 piece를 다운로드하고 있는 상태일 경우 (즉, 모든 piece를 소유하고 있지 않을 때) leecher라고 하며, 파일의 다운로드가 끝나고 완전한 사본을 소유하고 있을 때 (즉, 모든 piece를 소유하고 있을 때) seed라고 한다. 한 peer는 위 두 가지 상태중 하나의 상태에만 있을 수 있다.

2.1.6 Rarest-First Algorithm

어떠한 piece를 요구할 지를 결정하는 선택 기법이다. 각각의 peer는 자신과 연결된 peer가 어떠한 piece를 소유하고 있는지에 대한 리스트를 가지고 있고, 이 리스트를 바탕으로 가장 개수가 적은 piece를 선택하고 요청하게 된다. 이때, 각 peer가 소유한 정보를 바탕으로 선택하기 때문에 local rarest-first algorithm이라고도 한다.

2.1.7 Peer 선택 기법 (Choking Algorithm)

본 기법은 choking algorithm이라고도 불리는 기법으로, 자신이 업로드 할 peer를 선택하는 것으로 BitTorrent에서는 tit-for-tat 전략에 의해 peer를 선택하게 된다. tit-for-tat 전략이란 자신에게 각 peer가 업로드한 비율을 비교해서 가장 많이 업로드한 4개의 peer를 선택하는 전략이다. 주기적으로 호스트는 자신에게 upload해준 peer들의 upload rate를 계산하여 unchoke할 peer와 choke할 peer를 선택하게 된다. 만약 이러한 기법이 Bitorrent에 존재 하지 않으면 자신이 가지고 있는 piece를 다른 peer에게 전송해줄 이유가 없게 되고, 다른 peer에게 자신이 가지고 있는 piece를 업로드 하지 않은 채 자신이 받고자 하는 파일만 받고 바로 나가버리는 free-rider 문제가 심화된다[3].

2.2 BitTorrent 동작 과정

본 장에서는 BitTorrent 프로토콜을 이용해서 파일을 공유하는 세부 동작 과정을 설명한다. 그림 1은 BitTorrent의 주요 동작과정을 묘사한 것이다.

파일 배포자는 파일의 정보를 담은 metainfo file (즉 “.torrent” 파일)을 생성한다. 생성된 .torrent 파일은 tracker와 사람들이 다운로드 받을 수 있도록 TPB[9]나 Mininova[10]와 같은 잘 알려진 웹사이트에 등록한다. 이 파일에 관심을 갖고 있는 호스트는 해당 웹사이트로부터 “.torrent” 파일을 받는다 (그림 1의 A). 해당 호스트는 “.torrent ” 파일에 포함되어 있는 tracker의 주소로 그림 1의 B와 같이 접속을 한다. 이 때, tracker는 자신이 관리하고 있는 해당 파일을 공유하는 peer들 중에서 일부의 peer를 선택하여 그들의 주소가 담긴 리스트를 호스트에게 알려준다 (그림 1의 C). 이때, 보통 50개의 peer정보를 호스트에게 알려준다. 호스트는 tracker로부터 획득한 peer들의 주소를 바탕으로 연결을 시도하고 연결된 peer에 한해서 자신이 원하는 파일의 각기 다른 piece를 요청하게 된다 (그림 1의 D).

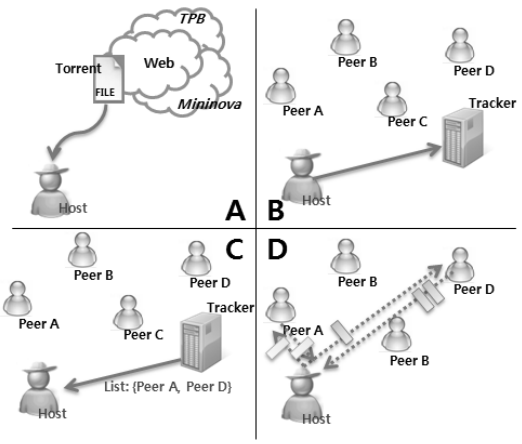

그림 1. BitTorrent 동작과정 모식도

Ⅲ. 테스트베드 설정

본 논문에서 unchoked peer 개수의 효과를 검증하기 위해 그림 2와 같이 서울대학교 캠퍼스 망에 테스트베드를 설치하였다. 본 테스트베드는 총 9개의 호스트로 구성되어 있고, 이중에서 8대는 파일을 다운로드 받고자 하고 leecher이고, 나머지 1대는 공유할 파일을 미리 갖고 있는 seed 이다. 또한 실

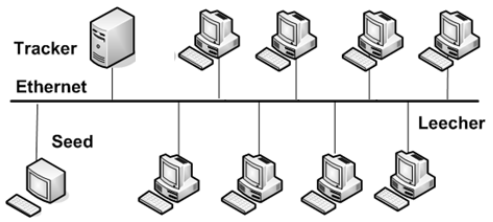

그림 2. 테스트베드 모식도

표 1. 실험 환경 설정

항목	값
파일의 크기	298Mb
Seed의 개수	1 개
Seed의 업로드 Peer 개수	2 개
Leecher의 개수	8 개
대역폭	100Mbps

험을 위해 테스트베드에 하나의 독립된 tracker도 설치하여 독립적인 실험을 수행할 수 있도록 하였다. 각 호스트에서 동작할 BitTorrent 클라이언트는 현재 많이 사용되고 있는 오픈 소스 기반의 Azureus[11]이다. 각 호스트의 상태를 매 시간마다 측정 및 기록하고, unchoked peer의 개수를 변화시키기 위해서 Azureus 클라이언트의 소스를 고쳐서 사용하였다. 실험을 위한 자세한 환경 설정은 표 1에 나타나 있다.

Ⅳ. 성능 평가 및 분석

본 장에서는 unchoked peer가 swarm의 상황에 따라 BitTorrent 성능에 어떤 영향을 끼치는지 분석하기 위해 다양한 상황을 가정하고 실험하였다.

Swarming system에서 성능을 가장 쉽고 정확하게 알 수 있는 방법은 swarm에 참여하는 각각의 peer들의 다운로드 속도를 모두 측정하여 비교 분석하는 것이다. swarm에 참여하는 peer들의 평균적인 다운로드 속도가 상승된다면 그것은 바로 swarm의 성능 향상을 의미한다. 따라서 본 실험에서는 swarm의 성능을 판단하기 위한 기준으로 다운로드 속도를 측정하고 분석하였다.

4.1 Swarm내 모든 peer의 unchoked peer 개수가 동일할 때

첫 번째 실험은 swarm내에 모든 peer가 동일한 unchoked peer개수를 가지고 파일을 공유할 때 효율

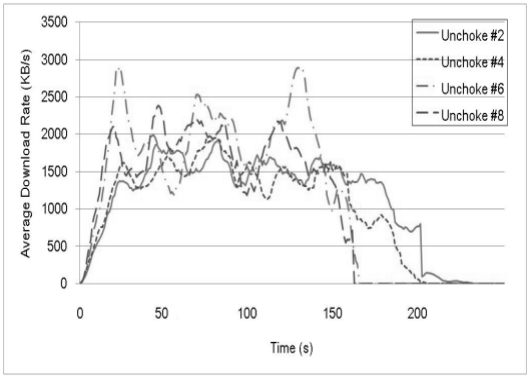

그림 3. 8개 호스트의 unchoked peer 개수가 각각 2개, 4개, 6개, 8개 일 때의 Average Download Rate

적인 unchoked peer 개수를 알아보기 위한 것이다.

그림 3에서 swarm내의 모든 peer들의 unchoked peer 개수가 각각 2, 4, 6, 8개 일 때 시간에 따른 각 peer의 다운로드 속도를 각각 나타내고 있고, 표 2에서 이에 따른 평균 파일 다운로드 종료시간을 나타내고 있는데, unchoked peer 개수가 8개 일 때, 2개일 때와 4개일 때 보다 각각 34%, 21% 종료시간이 단축 되었다. 즉 unchoked peer 개수가 늘어남에 따라 전체 swarm에서의 peer간의 업로드의 수와 양이 늘어나 되고, 이에 따라 swarm 전체적으로 다운로드 속도가 빨라지게 된 것이다. 하지만 자신을 제외한 전체 leecher개수만큼 unchoked peer 개수가 늘어 갈수록 효과는 줄어들었다. 그러므로 전체 swarm측면에서는 대역폭이 충분할 때, unchoke 개수가 늘어날수록 swarm 의 전체적인 성능이 향상됨을 알 수 있다.

4.2 Swarm내 peer의 unchoked peer 개수가 다를 때

다음 실험은 한 swarm내에 동일하지 않은 unchoked peer 개수를 지닌 호스트들이 파일을 공유할 때의 성능 차이를 알아보기 위한 것이다. 각각의 호스트들은 서로 다른 상황에서 파일을 공유하고 있고 (예: 높은 대역폭 사용 vs. 낮은 대역폭 사용), 또한 각자 swarm에 업로드로 공헌하려는 정도가 모두 다르다. 따라서 4.1장의 결과처럼 모든 호스트들이 많은 unchoked peer 개수를 사용하는 것은 전체 swarm 성능에 도움이 되지만 일부 호스트들은 많은 unchoked peer 개수를 사용하려 하지 않을 수 있다. 따라서 본 실험에서는 이와 같은 상황을 가정해서 unchoked peer 개수가 2개인 4개의 호스트와 unchoked peer 개수가 5개인 4개의 호스트가 존재

할 때 시간에 따른 다운로드 속도를 측정하였고, 또한 자유도를 높여서 두 호스트씩 짝을 지어서 unchoked peer 개수가 2개, 4개, 6개, 8개일 때의 실험을 진행하였다.

4.2.1 Unchoked peer 개수가 2개일 때와 5개일 때의 다운로드 속도 비교

그림 4는 unchoked peer 개수가 2개인 호스트들과 unchoked peer 개수가 5개인 호스트들의 평균 다운로드 속도를 나타낸다. 그림 4에서 보여주듯이 unchoked peer 개수가 2개인 호스트들이 보다 빠른 다운로드 속도를 보여주었고, 표 3에서 볼 수 있듯이 약 30초 정도 빠른 파일 다운로드 종료시간을 나타내었다. 즉 한 swarm내에 다른 수의 unchoked peer 개수를 지닌 호스트들이 참여하고 있을 때 적은 개수의 unchoked peer 개수를 사용하는 호스트들이 더 큰 이득을 봄을 알 수가 있다. 이것은 같은 양의 대역폭을 더 적은 수의 peer에게 할당한 unchoked peer 개수를 2개로 설정한 호스트들이 각 peer에게 할당한 업로드 대역폭의 양이 unchoked peer 개수 5개 호스트들보다 크기 때문에 상대적으로 다른 peer로부터 선택될 확률이 크기 때문이다. 즉, 이와 같은 경우에는 많은 peer에게 적은 양으로 업로드 하는 것 (즉, 많은 unchoked peer 개수를 설정하는 것) 보다, 적은 peer에게 많은 양으로 업로드 하는 것 (즉, 적은 unchoked peer 개수를 설정하는 것) 이 개개의 호스트에게는 더 유리하다. 물론 4.1장에서 보듯이 개개의 호스트가 이러한 선택을 한다면 전체 swarm 성능에는 오히려 좋지 않을 수 있다.

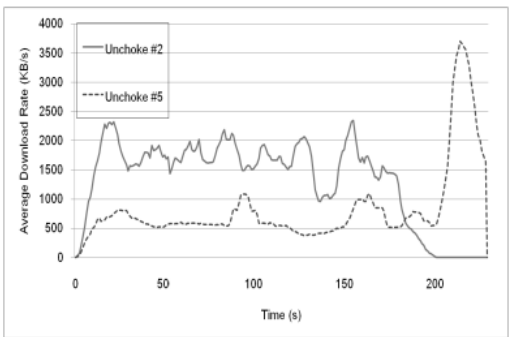

그림 4. Unchoked peer 개수가 2개인 호스트 4개와 5개인 호스트 4개의 Average Download Rate

표 3. Unchoked peer 개수에 따른 peer들의 평균 다운로드 종료시간

Unchoked peer 개수	평균 파일 다운로드 종료시간
2 개	201 초
5 개	228 초

4.2.2 Unchoked peer 개수가 2개, 3개, 4개, 5일때의 비교

마지막으로 좀 더 다양한 unchoked peer 개수를 가진 호스트들이 파일공유를 할 때의 효율적인 unchoked peer 개수를 알아보기 위해 2대씩 호스트를 묶은 4그룹의 unchoked peer 개수를 각각 2, 3, 4, 5씩 설정하고 실험을 하였다. 그림 5에서 보여주듯이 unchoked peer 개수가 상대적으로 작은 호스트들이 unchoked peer 개수가 큰 호스트들 보다 빠른 다운로드 속도를 나타내었고 그에 따라 표4에 나와있듯이 다운로드 종료 시간도 단축됨을 볼 수 있었다. 이것은 그림 4의 실험과 유사한 결과로써 한 swarm내에 각기 다른 unchoked peer 개수를 지닌 호스트가 존재할 때, unchoked peer 개수를 적게 설정 할 수록 호스트의 성능에 유리함을 보여준다.

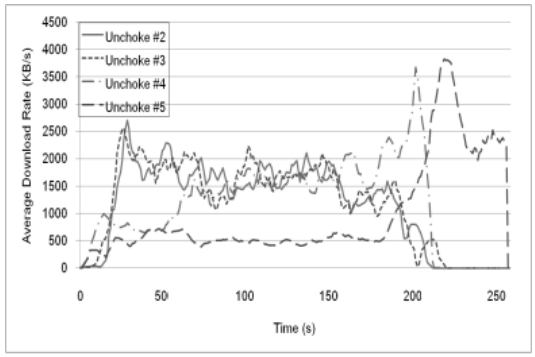

그림 5. 2개 호스트씩 무리를 지은 4개의 그룹이 각각 2개 3개 4개 5개의 unchoke 개수를 가질 때 Average Download Rate

표 4. Unchoke 개수에 따른 다운로드 종료시간

Unchoked peer 개수	평균 파일 다운로드 종료시간
2 개	211 초
3 개	219 초
4 개	220 초
5 개	225 초

Ⅴ. 결론 및 향후 과제

본 논문에서는 swarm 의 상황에 따라 unchoked peer 개수의 변화가 BitTorrent 성능에 영향을 주는 것을 보여 주었다. 전체적인 swarm의 성능을 고려하자면 unchoked peer 개수가 많을수록 전체 peer 들의 다운로드 속도와 종료시간에서의 이득을 볼 수 있었다. 하지만 각기 다른 unchoked peer 개수를 지닌 호스트들이 한 swarm에 참가할 경우 적은 unchoked peer 개수를 지닌 peer가 큰 unchoked peer 개수를 지닌 peer보다 성능향상에서의 이득을 보았다. 따라서 많이 변하는 swarm의 상황에 따라 전체적인 swarm의 성능과 각 호스트의 성능을 동시에 최대화 할 수 있는 unchoked peer 개수 선택 기법이 필요하다. 본 연구팀에서는 swarm의 상황에 따라서 unchoked peer 개수를 조정하는 적응적 unchoked peer 개수 선택 기법을 후속 연구로 진행하고 있다.

참 고 문 헌

〔1〕 B. Cohen. "Incentives build robustness in bittorrent." *In 1st Workshop on Economics of peer-to-peer Systems, 2003.*

〔2〕 Ipoque "The impact of p2p file sharing, voice over ip, instanct messaging, one click hosting and media streaming on the internet. http://www.ipoque.com/resources/internet-studies/internet-study-2008" 2009

〔3〕 Michael Sirivianos, Jong Han Park, Rex Chen, Xiaowei Yang "Free-riding in BitTorrent Networks with the Large View Exploit" In IPTPS 2007.

〔4〕 M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani. "Do incentives build robustness in bittorrent" In USENIX NSDI, 2007.

〔5〕 Arnaud Legout, G. Urvoy-Keller and P.Michiardi "Rarest First and Choke Algorithms Are Enough" In IMC 2006

〔6〕 A. Legout, N. Liogkas, E. Kohler, and L. Zhang. "Clustering and sharing incentives in bittorrent systems." In ACM SIGMETRICS, 2007.

〔7〕 D. Qiu and R. Srikant. "Modeling and performance analysis of bittorrent-like peer-to-peer networks." In ACM SIGCOMM, 2004.

〔8〕 A. Sherman, J. Nieh, and C. Stein. "Fairtorrent: bringing fairness to peer-to-peer systems." In ACM CoNEXT, 2009.

〔9〕 The pirate bay. http://thepiratebay.org

〔10〕 Mininova. http://www.mininova.org

〔11〕 Azureus - now called vuze - open source bittorrent client. http://azureus.sourceforge.net/.

정 태 중 (Taejoong Chung) 정회원

2009년 8월 POSTECH 컴퓨터공학과 학사

2009년 8월~현재 서울대학교 전기컴퓨터공학부 석사과정

<관심분야> Peer-to-Peer, Contents Distribution

한 진 영 (Jinyoung Han) 정회원

2007년 8월 KAIST(한국과학기술원) 전산학과 학사

2007년 9월~현재 서울대학교 전기컴퓨터공학부 박사 과정

<관심분야> Peer-to-Peer, 유비쿼터스 컴퓨팅

김 현 철 (Hyunchul Kim) 정회원

1995년 2월 KAIST(한국과학기술원) 전산학과 학사

1997년 8월 KAIST(한국과학기술원) 석사

2005년 2월 KAIST(한국과학기술원) 박사

2008년 3월~현재 서울대학교 전기컴퓨터공학부 BK 조교수

<관심분야> Internet Traffic Measurement, Contents Oriented Networking, Network Security

권 태 경 (Ted “Taekyoung” Kwon) 정회원

1993년 2월 서울대학교 컴퓨터공학과 학사
1995년 2월 서울대학교 컴퓨터공학과 석사
2000년 2월 서울대학교 컴퓨터공학과 박사
2002년 3월~현재 서울대학교 전기컴퓨터공학부 교수
<관심분야> 센서 네트워크, 유비쿼터스 컴퓨팅

최 양 희 (Yanghee Choi) 종신회원

1975년 2월 서울대학교 전자공학과 학사
1977년 2월 한국과학기술원 전자공학과 석사
1984년 2월 프랑스 ENST 전자학 박사
1991년 3월~현재 서울대학교 전기컴퓨터공학부 교수
<관심분야> 미래 인터넷, 멀티미디어 통신