다양한 P2P 응용을 위한 분산 해쉬테이블 기반 오버레이 네트워크

광주과학기술원 박찬모 · 김종원

1. 서 론

P2P(Peer-to-Peer)는 1999년 5월 Napster[1]를 시작으로 대중적인 인기를 갖게 되었으며 이후 다양한 서비스들이 P2P 네트워크를 기반으로 등장하게 되었다. 또한 국내에서도 소리바다[2]를 통한 MP3 파일 공유로 P2P에 대한 관심이 고조된 바 있다. 하지만 P2P는 일반적으로 인식되는 파일 공유와 같은 서비스만으로 국한되지 않는다. P2P의 정의는 매우 다양하며, 서버/클라이언트 방식과 대비하는 개념에서 정의되는 P2P는 참여자들이 가진 자원(CPU, 디스크, 네트워크 링크, 프린터, 파일 등)의 일부를 공유하는 분산 시스템으로 참여하는 모든 노드들이 서버와 클라이언트의 역할을 동시에 담당하는 것이다[4]. 이들 자원의 공유를 통한 P2P 서비스는 특정 작업을 위한 협업(collaboration) 서비스, 서비스될 데이터를 사용자에 가장 근접한 위치에서 제공하도록 하는 에지(edge) 서비스, 분산된 디스크나 CPU를 활용하기 위한 분산 컴퓨팅 및 자원(distributed computing and resource)의 공유 서비스, 그리고 각 컴퓨터에 있는 에이전트를 통해 동적으로 정보를 교환하며 특정 문제를 해결토록 하는 인텔리전트 에이전트 서비스(intelligent agent service) 등으로 분류할 수 있다[5].

P2P 시스템이 갖는 특성은 다음의 다섯 가지로 분류될 수 있다[5]. 먼저 네트워크 상에 분산된 노드들로 구성되어야 한다. 둘째로 분산된 노드들로 구성된 P2P 네트워크는 일반적으로 응용 프로그램 계층에서 생성되는 네트워크를 이용하며, 이를 오버레이(overlay) 네트워크라 한다. 셋째로 P2P 시스템에서 자원에 대한 권한은 노드에 있으며 중앙집중식 관리는 없어야 한다. 넷째로 모든 노드들은 클라이언트이면서 동시에 서버로써 역할을 수행하는 대칭적 기능을 수행할 수 있어야 한다. 마지막으로 각 노드는 이질성(heterogeneity)을 가지며 동적인 환경에서 동작한다. 이것은 각 노드가 P2P 시스템에 빈번하게 참여하고 이탈할 수 있음을 의미한다. 따라서 P2P 시스템은 고장에 대한 내구성과 자기 조직성을 가져야 한다.

상기한 바와 같이 ***P2P 시스템의 근간에는 오버레이 네트워크가 있으며, 이에 관한 연구들은 초기의 Napster나 소리바다와 같은 중앙집중식 연결을 지나서 완전하게 분산된 네트워크 연결에 집중되고 있다.*** 완전하게 분산된 P2P 시스템의 오버레이 네트워크는 노드간의 임의적인 연결을 통해 만들어지는지 또는 정의된 방식을 통해 체계적으로 형성되는지에 따라 비구조적 및 구조적 P2P 네트워크로 분류된다[6]. 비구조적인 P2P 네트워크는 Gnutella 등에서 활용된 바 있으며, 구조적인 P2P 네트워크는 대표적으로 분산해쉬테이블(distributed hash table: DHT, 이하에서는 DHT로 기술함) 기반으로 구현되고 있는 추세이다.

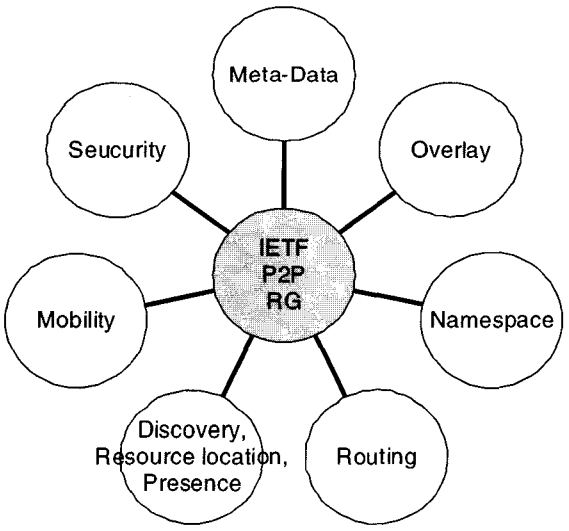


그림 1 P2P 연구 분야 관계도 (IETF P2P RG)

따라서 본 논문에서는 그림 1과 같이 2003년 조직된 IETF(Internet engineering task force) P2P 연구그룹[7]에서 정의하는 각종 연구 분야들 중에서 오버레이 네트워크를 중심으로 한 라우팅(routing)과 자원 위치 판별(resource location) 분야에 주목한다. 특히 구

조적인 P2P 오버레이 네트워크를 DHT 기반으로 생성하는 분야를 중심으로 현재까지의 P2P 연구 내용과 최신 동향을 다루고자 한다. 본 논문의 구성은 먼저 2절에서는 DHT 기반 오버레이 네트워크 구성에 대해 소개하고, 이에 대한 대표적인 연구들인 Chord, Pastry, Tapestry 그리고 CAN(content addressable network) 등에 대해 논의하고 서로 비교한다. 또한 2절의 마지막에서는 DHT 기반의 오버레이 네트워크의 형성시 네트워크 근접성을 통한 성능 향상을 시도하는 접근 방식을 소개한다. 이어서 3절에서는 DHT 기반의 오버레이 네트워크를 라우팅과 자원 위치 판별을 위한 하위 계층으로 이용하여 P2P 분산 스토리지 분야에 응용하고 있는 OceanStore, PAST 그리고 CFS (cooperative file system)을 간략히 소개한다. 마지막으로 4절에서 본 논문의 결론을 맺는다.

2. P2P 오버레이 네트워크를 위한 분산 해쉬 테이블 (DHT) 방식들

Gnutella와 같은 분산화 된 오버레이 네트워크를 이용한 P2P 파일 공유의 경우, 중앙 서버를 통한 공유의 경우와 달리 파일들에 대한 디렉토리 서비스를 일괄적으로 제공하지 못하기 때문에 전체 노드들에 검색 요청 메시지를 보내 응답을 기다려야 한다[6,8]. 또한 이때 중복 트래픽이 과도하게 발생하여 네트워크 트래픽 부하가 급속히 증가하게 된다. 또한 검색에 대한 응답의 불규칙성 및 검색 노드 제한에 따라 검색 결과의 완전성에도 문제를 갖게 된다. 이와 같이 비구조적인 P2P 네트워크를 형성하는 Gnutella와 같은 공유 방식의 문제를 해결하기 위해서 P2P 오버레이 네트워크 형성에 구조적인 특성을 부여하여 검색 효율성을 제고하는 방법이 필요하다. 그리하여 그림 2에 제시한 것과 같이 2001년부터 DHT를 이용한 P2P 오버레이 네트워크에 대한 연구들이 활발하게 진행되어 오고 있다. 본 논문에서는 2.1절에서 DHT에 대해 논의하고, 나머지 부분에서 이를 기반으로 하는 P2P 네트워킹에 대한 그간의 연구들에 대해 살펴본다.

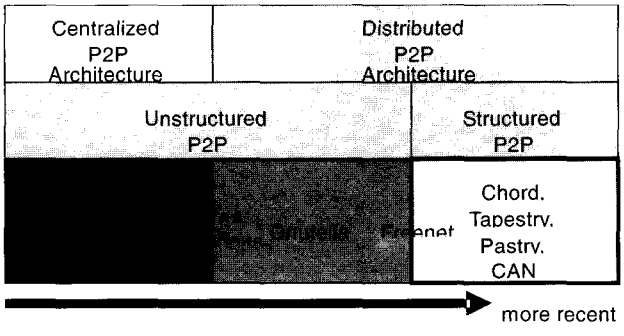


그림 2 P2P 응용 프로그램의 발전 추이 및 DHT 방식의 변천

2.1 분산 해쉬테이블(DHT)

해쉬테이블은 실제 값과 이에 해쉬함수(hash function)를 적용하여 형성한 키(또는 식별자)의 조합에 대해 배열을 사용하여 검색을 빠르게 하기 위한 자료 구조이다. 이를 네트워크 환경에 위치한 노드들에 분산하여 적용한 것이 DHT(분산해쉬테이블)이다. DHT 개념 이해를 돕기 위해 파일 공유 P2P를 예로 들어 보자. DHT를 사용하는 가장 큰 목적은 오버레이 네트워크에 구조화를 촉진하여 신속하고 체계적인 검색 및 라우팅이 수행하고자 하는 것이다[13]. 예를 들어 파일 공유의 경우에는 공유 네트워크를 구성하는 노드들에 DHT 버킷(bucket)을 분산시킨 다음, DHT 버킷을 찾기 위해 버킷을 가진 노드를 쉽게 찾을 수 있도록 하는 것이다. 따라서 DHT 버킷과 노드를 연결시키기 위한 방법이 필요하며, 일반적으로 노드들은 노드 식별자를 사용하여 버킷을 나타내고 각 파일들을 버킷과 연결하기 위해 파일 이름을 해쉬한 식별자를 사용한다. 이 경우 노드 식별자와 해쉬된 파일 이름은 동일한 크기의 이름 공간 (bit 수)을 가진다. 한편 DHT 기반 오버레이 네트워크에서는 노드 식별자를 사용하여 한 노드가 연결하는 노드(이웃 노드)들이 결정된다. 이는 DHT 기반 오버레이 네트워크에서 전체 노드들의 구성이 체계적으로 구조화됨을 의미하며, 이런 이유로 DHT 기반 P2P 네트워크를 구조적이라 하는 것이다.

이와 같은 DHT 기반 오버레이 네트워크를 활용하여 파일을 공유하는 과정은 크게 1)파일 등록, 2)파일 검색, 그리고 3)메시지 라우팅의 세가지로 구분된다.

1) 파일 등록 : 가지고 있는 파일을 공유하기 위해 DHT에 등록하는 것이며, 해쉬테이블의 Insert()와 유사한 동작을 한다. 차이점은 해쉬테이블 버킷이 분산되어 있으므로 이를 찾기 위한 과정이 필요하다는 것이다. 먼저 등록하고자 하는 파일의 이름을 사용하여 어떤 노드(버킷)에 등록할 것인지를 결정해야 하며, 이때 해쉬함수가 사용된다. 파일 이름을 해쉬한 값은 목적지 노드(버킷)에 대한 정보를 제공한다. 등록 메시지와 함께 파일 이름을 해쉬한 값을 보내면 라우팅에 따라 목적지 노드에 전달되어 저장된다.

2) 파일 검색 : 대상 파일 이름을 사용하여 목적지 노드에 대한 정보를 얻은 후 이를 검색 메시지와 함께 보내어 파일을 실제 소유하고 있는 노드의 정보를 얻는 것으로 Lookup()과 유사하다. 검색 메시지가 목적지 노드에 전달되기 위해서는 메시지 라우팅이 필요하다.

3) 메시지 라우팅 : 메시지에 포함된 목적지 노드 정보를 사용하여 메시지를 노드에 전달하는 것이다. 노

드는 이웃 노드들에 대한 라우팅 테이블을 가지며 메시지를 목적지 정보와 이웃 노드들의 노드 식별자를 비교하여 근접한 이웃 노드로 메시지를 전달한다.

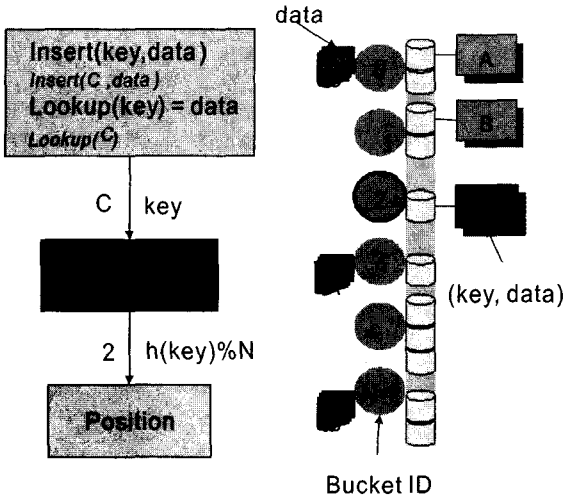


그림 3 DHT의 개념도 (insert와 lookup)

참고로 그림 3에 Insert와 Lookup 함수가 해쉬함수와 함께 사용되며, 키(key)를 이용하여 파일 이름 "C"인 파일을 DHT 상에 Insert 하고 Lookup 하는 예를 보여주고 있다. 여기서 data는 파일 내용이나 파일을 가진 현재 노드에 대한 정보를 의미한다. 이러한 개념을 공유하면서 개발된 DHT 기반 P2P 시스템들을 이후부터 자세히 살펴볼 예정이며, 이들 각종 DHT 간의 차이점은 식별자의 이름 공간을 구성하는 방법과 메시지 라우팅 알고리즘에 있다는 것에 착안하여 검토하기 바란다[6].

2.2 Chord

Chord는 MIT와 U.C. Berkeley 대학에서 2001년 제안한 DHT 기법으로 노드와 키의 해쉬 값을 위해 그림 4와 같이 원형의 m-bit 식별자 공간을 사용한다[9]. 이것은 consistent hashing을 분산 시스템에 적용하기 위해 변형한 경우에 해당된다. 키(key)를 해쉬한 값을 식별자 공간으로 대응시킬(mapping) 때는 consistent hashing과 같이 원형 식별자 공간에서 해쉬된 키 값이 같거나 더 큰 노드 식별자를 갖는 첫 번째 노드에 저장한다. 이 노드를 successor 노드라 한다. 또한 분산 환경에 적용시키기 위해 라우팅 테이블로 finger table을 각 노드가 유지하도록 한다. 이것은 consistent hashing에서 요구되는 전체 네트워크 상의 노드에 대한 정보를 분산된 동적인 환경에서 효과적으로 만족시키기 위해서 도입되었다. Finger table은 키를 삽입하거나 키를 가진 노드를 찾기 위한 lookup 등의 메시지를 해당 노드로 전달하는데 이용된다. Finger table은 유지 정보를 최소화하기 위해 원형 식별자 공간에서 노드의 위치(p)를 기준으로 하여 시계 방향으로 거리를 지수적으로 증가시켜 가면서 자신의 식별자보다 적은 식별자 값을 가진 노드들에 대한 정보를 갖는다(p+2i). 각 노드 정보는 범위(interval)와 범위의 시작 값(start) 그리고 시작 값에 대한 successor 노드를 가진다. 또한 오버레이 네트워크의 유지를 위해 이전 노드(predecessor)에 대한 정보를 유지한다. 메시지 라우팅은 각 노드의 finger table에 따라 범위를 지수적으로 감소시키면서 목적지 노드를 찾게 된다. 그림 4는 3 bit로 구성된 환형 ID 공간과 각 노드가 가지는 finger table, 또한 노드 3에서 finger table에 있는 노드의 범위를 보여준다. 라우팅의 예로 노드 3에서 키 1에 대한 포인터를 가진 노드를 찾기 위해 메시지가 전달되는 과정을 보여준다.

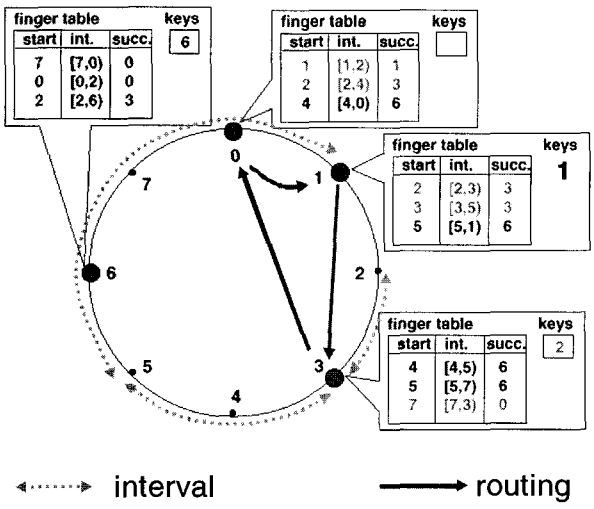


그림 4 Chord에서 노드 finger table과 라우팅 (노드 3에서 키 1을 lookup 하는 경우)

2.3 Pastry

Pastry는 2001년 Rice 대학과 Microsoft에 의해 제안된 DHT 기법이다[10]. 각 노드는 오버레이 네트워크에 조인할 때 IP 주소나 공개 키를 암호화 해쉬를 통해 중복되지 않는 랜덤한 128비트 노드 식별자로 만듦에 의해 노드 식별자들이 이름 공간상에 분산되게 한다. 또한 데이터도 128 비트 키를 가지며, 데이터로부터 유추되는 키에 가장 가까운 키를 가진 노드에 저장된다. 각 Pastry 노드들은 메시지의 전달을 위해 neighborhood 집합 (이웃 노드들 중 네트워크 상 가까운 노드들), leaf 집합 (이웃 노드들 중 식별자의 값이 가장 가까운 노드들), 그리고 라우팅 테이블 (이웃 노드들)등을 갖는다. Pastry 기법에서 라우팅 알고리즘의 기본 개념은 그림 5와 같다[21]. 노드들을 주어진 식별자 공

간을 prefix를 조합하여 동일한 prefix를 가진 노드들을 그룹화하고 이를 계층화한다. 가장 내부에 있는 그룹의 노드들은 그룹내의 모든 노드들에 대한 정보와 함께 상위 그룹에 있는 한 개 노드에 대한 정보를 라우팅 테이블에 갖는다. 메시지의 라우팅은 메시지에 있는 목적지와 가장 긴 동일 prefix를 가진 노드로 메시지를 전달하게 하여 목적지 노드에 도착하도록 한다.

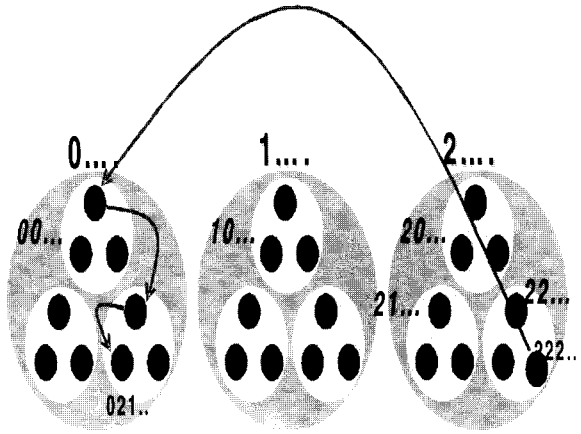


그림 5 Pastry에서 라우팅 기본 개념 (노드 222..에서 키 021... lookup)

2.4 Tapestry

Tapestry는 2000년 Berkeley 대학에서 제안한 DHT 기법이다[11]. 중앙집중된 리소스를 갖지 않고 point-to-point 링크만을 사용하여 객체나 서비스 중 가장 가까운 위치의 사본으로 메시지를 직접 위치 독립적으로 라우팅하는 위치 검색과 라우팅을 위한 infrastructure로 제안되었다. 노드와 객체들은 160 bit 식별자들이 해쉬함수를 통해 할당되며 메시지들은 목적지 식별자에 따라 가장 긴 suffix를 공유하는 노드로 라우팅된다. 검색하고자 하는 객체를 가진 노드의 위치는 객체 식별자와 동일한 노드 식별자를 가진 노드(root node)로 라우팅을 해서 알 수 있다. 만약 동일한 ID를 가진 노드가 존재하지 않는다면 최소한 하나의 digit가 동일한 노드로 라우팅된다. 각 노드들은 동일 suffix 수에 따라 레벨을 갖는 neighbor map을 갖는다. 그림 6에서 L1, L2, L3, 그리고 L4들은 레벨을 가리킨다. 또한 새로 참여하는 노드의 neighbor map 생성을 돕고 네트워크 유지를 위해 자신을 이웃 노드로 하는 노드들에 대한 역 포인터 리스트를 갖는다.

2.5 CAN

CAN(content addressable network)은 2001년

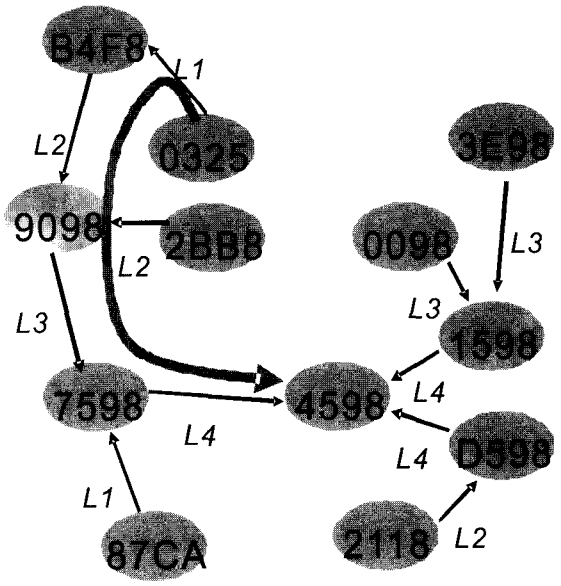


그림 6 Tapestry 라우팅 기본 개념 (노드 0325에서 키 식별자 4598을 Lookup)

ACIRI(AT&T Center for Internet Research)와 Berkeley 대학에서 빠른 검색을 지원하기 위해 제안한 DHT 기법이다[12]. CAN에서 데이터는 유일 키를 해쉬함수를 통해 d 차원에서 한 점에 대응되는 벡터 P (p=hash(key))를 얻는다. CAN은 d 차원 torus 상에 d 차원 가상 공간을 가지며 이 가상 공간은 많은 d 차원 존들로 분할된다. 각 노드들은 하나의 존에 대응되며 해쉬함수를 통해 가상 공간을 분할한 존의 범위에 해당하는 데이터를 저장한다. 두 개 노드가 속한 좌표 값 중 d-1 차원의 값이 중첩되며 한 개 차원이 이웃하면 이웃 노드라 한다. 각 노드들은 이들 이웃 노드들의 존 정보와 IP 주소를 가진 라우팅 테이블을 갖는다. 그림 7은 2차원 좌표 공간상의 CAN 공간 분할 및 노드 8의 조인 결과를 보여준다. 노드 1의 이웃 노드들은 현재 표시된 노드에 따르면 {2,3,4,7}, 노드 3의 이웃 노드는 {1,6} 그리고 노드 6의 이웃 노드들은 {2,3}이다. 노드 6은 노드 8번의 조인 메시지를 노드 4로 라우팅한다. 라우팅은 좌표 공간상에서 목적지인 4와의 거리를 최대한 감소시키도록 이웃 노드들 선택하여 메시지를 포워딩한다. 노드 4는 현재 가진 존을 X나 Y축의 영역을 이분하는 것으로 존을 노드 8에 할당한다.

2.6 DHT 방식 비교

본 절에서는 지금까지 살펴본 DHT들의 특징과 이를 활용하는 P2P 시스템들을 성능, 확장성, 신뢰성, 유지보수성 그리고 사용의 편의성 등의 비교를 시도한다[13]. 표 1에 제시한 것과 같이 성능의 측면에서

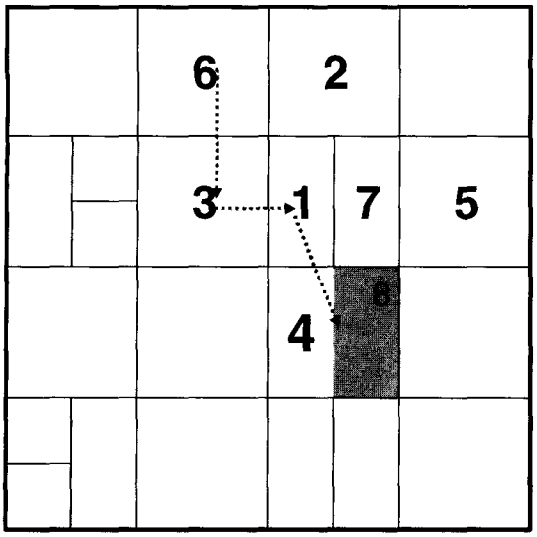


그림 7 2차원 CAN에서의 노드 8의 조인 과정 예

Chord, Pastry, Tapestry 등은 lookup hop과 노드 추가시 O(logN)의 비슷한 성능을 보여준다. 여기에서 N은 ID 공간의 크기이다. 반면 CAN의 성능은 O(N1/d) 이며, 여기에서 d는 차원 공간 값이며 N은 실제 노드의 수이다. CAN의 경우 차원이 증가함에 따라 성능이 개선되므로 이들 모두는 좋은 성능을 가진다고 할 수 있다. 또한 확장성은 노드의 수가 증가함에 따라 유지해야 하는 라우팅 테이블의 크기와 lookup 성능으로도 평가할 수 있다. Chord, Pastry, 그리고 Tapestry 등은 O(logN)이다. 하지만 CAN에서 라우팅 테이블 크기는 가상 공간의 차원 값이므로 라우팅 테이블의 크기 측면에서 보다 좋은 확장성을 가진다고 할 수 있다. 신뢰성은 노드가 연결 해제되었을 때 전체 시스템에 끼치는 영향을 고려한다. 이 경우 CAN은 우회 라우팅이 존재하여 모두 recovery 메커니즘을 가지므로 좋은 신뢰성을 가진다. 유지 보수성은 네 가지 시스템 모두 동적인 노드의 추가/삭제가 가능하므로 좋다고 할 수 있다. CAN의 경우 유지 보수성은 라우팅 테이블의 수에만 영향을 미치므로 가장 특성이 좋다.

2.7 DHT 기반 오버레이 네트워크의 효율성 향상

이와 같은 DHT에 근간한 오버레이 네트워크의 효율성을 향상하기 위해서는 형성된 오버레이 네트워크가 실제 네트워크 환경의 상황을 반영하여 근접성(physical proximity)을 향상할 필요가 있다. DHT는 (key, value) 조합을 단일 서버에서 사용하는 해쉬테이블과 같이 저장한다. 이를 이용하여 빠른 검색을 가능케 하며 추가로 적은 라우팅 홉을 가지고 메시지가 전달되도록 한다. 또한 DHT에서 노드들은 자기 조직화 되며 참여 노드들에 대한 작은 라우팅 테이블을 유지한다. DHT에서 서로 다른 특성은 메시지를 라우팅하는 알고리즘이다. 메시지 라우팅 알고리즘에 따라 다음 노드로 전달할 때, 메시지를 수신한 노드가 송신한 노드 사이의 실제 네트워크의 근접성을 반영하는 것이 중요하다. 이를 반영하기 위한 연구들은 다음과 같은 방향으로 진행되고 있다[17].

표 1 P2P 오버레이 네트워킹을 위한 DHT 방식 비교표

	Chord	Pastry	Tapestry	CAN
Size of Routing Table	O(logN)	O(logN)	O(logN)	2d
# of Hops on lookup	O(logN)	O(logN)	O(logN)	$O(N^{1/d})$
Node Insert	O(logN)	O(logN)	O(logN)	$O(N^{1/d})$
Locality	No	Yes	Yes	No
Load Balance	-	No	No	Yes
Performance	Good	Good	Good	Good
Scalability	Good	Good	Good	Good
Reliability	Good	Good	Good	Good
Maintenance	Good	Good	Good	Best
Usability	Simple	Simple	Good	Simple

1) Geographic layout[18] : 노드 ID를 할당할 때 실제 네트워크에서 가까운 노드들은 가까운 노드 ID가 할당되도록 한다. 즉 실제 네트워크 위상을 ID 공간에 맵핑하기 위한 것이다. CAN에서 라우팅 성능 개선을 위해 사용되어 왔으며 현재는 Chord에서 연구가 진행되고 있다. 이것은 CAN에서 ID 공간을 d-차원의 사용이 가능하기 때문에 적용이 용이하다.

2) Proximity routing[12] : 각 노드의 라우팅 테이블 네트워크 상 거리를 고려하지 않고 만들어진다. 메시지를 라우팅할 때 라우팅 테이블에 있는 노드들 중 네트워크 상에 가장 가까운 노드를 라우팅 알고리즘에 따라 선택하도록 하는 것이다. CAN에 대해 제안되었으며 Tapestry, Pastry 그리고 Chord에서도 사용될 수 있다. 여기에서 Tapestry와 Pastry에서 적용되면 많은 후보 노드들 중 선택이 가능하므로 좀 더 뚜렷한 성능 향상을 기대할 수 있다.

3) Proximity neighbor selection[20] : 라우팅 테이블을 생성할 때 각 엔트리들을 네트워크 거리를

고려하여 선택한다. 초기 라우팅 테이블 생성시 지역성을 반영할 수 있기 때문에 Tapestry와 Pastry에서 사용된다.

3. DHT 기반 오버레이 네트워킹을 활용한 P2P 응용 프로그램들

P2P 응용 프로그램은 피어 컴퓨터들로 구성된 오버레이 네트워킹에 근간하여 지속성을 제공한다. 일반적으로 P2P 네트워크에 참여하는 노드 컴퓨터들은 데이터를 저장하는 서버 기능, 메시지를 라우팅하는 라우터 기능, 라우터를 통해 서버에 데이터를 저장하거나 검색하는 클라이언트 기능, 데이터를 캐쉬하는 캐쉬 기능 등의 다양한 역할을 수행한다. 또한 P2P 응용 프로그램들은 대부분 신속성, 오류에 대한 견고성, 확장성, 그리고 신뢰성 등을 필요로 한다. 따라서 연구자들은 오버레이 네트워크를 인터넷 라우팅에서부터 분산된 네트워크 스토리지 분야에 걸친 다양한 응용에 적용하고 있다. 특히 논문에서는 DHT 기반 오버레이 네트워킹을 활용하는 P2P 응응 프로그램들 중에서 스토리지 응용에 주목하여 P2P를 이용하는 OceanStore[14], PAST[15], 그리고 CFS[16] 등을 살펴본다. 이들은 모두 라우팅과 위치 판별을 DHT 기반 오버레이 네트워킹으로 수행하면서 스토리지 및 캐쉬 관리 기능을 추가한 응용 프로그램들이다.

3.1 OceanStore

OceanStore는 2001년 U.C. Berkeley에서 개발중인 분산 P2P 스토리지이며 Tapestry DHT에 근간한 라우팅 및 위치 판별을 하는 오버레이 네트워크를 사용한다[11]. OceanStore는 이동성 데이터 지원을 위한 스토리지 관리 시스템으로 서로 신뢰성이 없는 노드들로 구성된다[14]. OceanStore에서 노드, 서버, 그리고 라우터 등의 객체들을 모두 GUID(globally unique identifier)를 사용하여 구분한다. 객체들을 액세스하는 클라이언트는 목적지 ID로 GUID를 사용한다. 객체들은 여러 개의 서버에 복제되고 저장된다. 이를 floating replicas라 하며 여러 버전의 객체들이 저장될 수 있다. 객체는 가장 최근의 "active" 버전과 이것을 제외한 "archival" 사본으로 분류된다. OceanStore는 "read/write"에 대한 접근 제어를 지원한다. 객체들은 update를 통해서만 수정이 가능하며 이때 새로운 버전이 생성되며 일관성은 버전을 통해 유지된다.

3.2 PAST(Persistent Active STorage Service)

PAST는 2001년 Microsoft에 의해 개발되기 시작한 P2P 스토리지 응용이며, 라우팅 및 위치 검색에 cooperative 웹 캐싱, 단체 통보(group notification), 인스턴트 메시징(instant messaging) 분야에도 적용된 바 있는 Pastry 기반의 DHT를 이용하고 있다[15]. PAST는 시스템에 파일 저장과 검색(retrieve)을 요청할 수 있으며 이들 요청들을 라우팅할 수 있는 인터넷 노드들로 구성된다. 또한 노드들은 자신의 스토리지를 PAST에 제공할 수도 있다. PAST 노드들은 오버레이 네트워크에 자기조직화(self-organizing) 할 수 있다. 클라이언트에 의해 PAST에 저장되는 파일들은 다중 노드에 복제되어 영속성 및 가용성이 보장된다. 인기있는 파일들의 사본은 쿼리 로드 밸런스를 위해서 PAST 노드에 캐쉬하며, 노드의 다수성과 다양성을 통해 백업이나 보관 데이터 보호의 필요성을 제거했다. PAST에 저장된 파일들은 파일 저장시에 할당되는 fileId를 가지므로 동일한 fileId를 가진 파일들은 존재하지 않는다. 이것은 파일 변경이 불가능하다는 것이다. 또한 delete 오퍼레이션을 제공하지 않고 대신에 파일 대체 오퍼레이션만을 제공한다. 클라이언트의 요청들은 Pastry 라우팅에 따라 네크워크에서 현재 요청 노드에 가장 근접한 노드로 전달되도록 한다. 보안 쿼타 시스템을 통해 각 노드에서 스토리지 제공과 요구를 조절하며 이것은 브로커를 통해 이루어진다. 모든 노드는 브로커를 신뢰하지만 사용자들은 파일을 저장하거나 검색(retrieve)할 때 자신을 숨길 수 있게 되어 있다.

3.3 CFS(Cooperative File System)

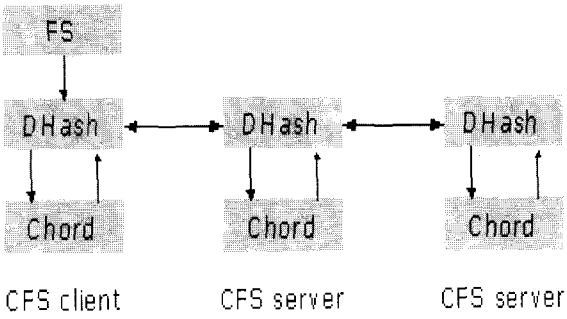


그림 8 CFS 서버/클라이언트 구조

CFS(Cooperative File System)은 2001년부터 MIT에서 Chord를 기반으로 하여 개발중인 P2P 스토리지 시스템이다[16]. CFS은 효율적이고 안정적이며 로드밸런싱을 지원하는 파일 저장 및 복구를 위한 읽기 전용 스토리지 시스템이다. 그림 8에 제시한 바와 같이 CFS 서버는 블록 스토리지를 위한 DHT(일명 DHash)를 제공하며 클라이언트는 DHash 레이어 블록들로 저장된 파일 시스템 데이터와 메타 데이터(디렉토리 데이

터)등을 변환하여 응용 프로그램에 읽기 전용 파일 시스템 인터페이스를 제공한다. DHash 레이어는 로드밸런싱을 위해 수 킬로 바이트 크기의 블록으로 분산하고 캐싱하며 안정성을 위해 복제하고, 서버를 선택할 수 있도록 하여 지연을 감소시킨다. DHash는 Chord 위치 검색 프로토콜을 사용하여 블록을 검색한다. DHash 레이어에서 파일 시스템은 블록들로 나누어지며 이들 블록들은 해쉬함수를 통해 얻어진 128-bit ID의 Successor 노드에 Chord 로케이션 알고리즘에 따라 저장되며 복제는 파라메터로 정해지는 수만큼 successor 노드의 다음 successor 노드들에 복제된다.

4. 결 론

지금까지 비구조적인 P2P 네트워크에서 라우팅과 리소스 위치 검색 문제를 해결하기 위해 DHT 기반 오버레이 네트워크를 중심으로 개발된 P2P 시스템들에 대해 살펴보았다. 이들은 원형 1차원 ID 공간을 사용하는 Chord와 suffix나 prefix 매칭을 통한 Tapestry와 Pastry, 그리고 다차원 ID 공간 사용을 통한 CAN 등이다. 이들은 공통적으로 각 노드는 ID 공간을 분할하여 해당 부분에 대한 디렉토리 서비스와 같은 기능을 담당한다. 차이점은 ID 공간 구성의 차이에 따른 메시지의 라우팅 알고리즘 등의 변화가 대표적이다. 또한 대부분의 DHT 기반 P2P 시스템들은 실제 네트워크 상황을 반영하지 않고 오버레이 네트워크를 단순하게 사용하는 문제점을 개선하기 위한 최신 연구들도 소개하였다. 이어서 DHT 기반 오버레이 네트워킹을 활용하는 각종 P2P 스토리지 응용 프로그램들에 대해 살펴보았으며, 이들이 DHT를 기반으로 구조적인 P2P 네트워크를 형성하고 이를 이용하여 효율적인 분산 스토리지 시스템을 제공하고자 함을 확인할 수 있었다. 마지막으로 향후에도 오버레이 네트워크의 향상을 위한 다양한 시도들이 예를 들어 네트워크 운영 비용을 최소화하기 위해서, 네트워크 고장에 대한 강인성을 향상시키기 위해서, 보안성을 증대시키기 위해서, 그리고 실제 네트워크의 상황을 보다 적절하게 반영하기 위해서 계속되어 P2P 응용의 발전을 도모할 것으로 기대된다.

참고문헌

[1] Napster, http://www.napster.com/.

[2] 소리바다, http://www.soribada.com/.

[3] Nelson Minar, "Distributed systems topologies part 1-2," http://www.openp2p.com/pub/a/p2p/2001/12/14/topologies_one.html, http://www.openp2p.com/pub/a/p2p/2002/01/08/p2p_topologies_pt2.html, Dec. 2001.

[4] R. Schollmeier, "A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications," in Proc. IEEE Conference P2P 2001, Linköping Sweden. August 2001.

[5] Internet2 P2P Working Group, http://p2p.internet2.edu.

[6] S. Androutsellis-Theotokis, "A survey of peer-to-peer file sharing technologies," Technical Report WHP-2002-03, Athens University of Economics and Business, 2002.

[7] IETF P2P Research Group, http://www.irtf.org/charters/p2prg.html.

[8] M. Jovanovich, F. Annexstein, and K. Berman, "Scalability issues in large peer-to-peer networks - a case study of gnutella," Technical Report of ECECS Dept., Univ. of Cincinnati, 2001.

[9] I. Stoica, R. Morris, D. Karger, F. Kaasho ek and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for Internet applications," in Proc. ACM SIGCOMM 2001.

[10] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in Proc. IFIP/ACM International Con-ference on Distributed Systems Platforms (Middleware), Nov. 2001.

[11] B. Zhao, J. Kubiatowicz, and A. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," Technical Report UCB/CSD-01-1141, Computer Science Division, Univ. of California, Berkeley, April 2001.

[12] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in Proc. of ACM SIGCOMM, 2001.

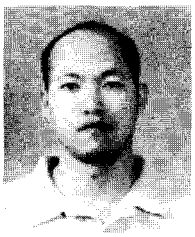
[13] F. Harrell, Y. Hu, G. Wang, and H. Xia, "Survey of locating and routing in peer-to-peer systems," http://www.sics.se/~sameh/research/P2P/Surveys and Comparisons/Surveyof Locating and Routing in

p2p systems/group15.pdf.
[14] OceanStore, http://oceanstore.cs.berkeley.edu/.
[15] PAST, http://research.microsoft.com/~antr/PAST/default.htm.
[16] CFS (Cooperative File System), http://www.pdos.lcs.mit.edu/papers/cfs:sosp01/.
[17] M. Castro, P.Druschel, Y. C. Hu, and A. Rowstron, "Exploiting network proximity in peer-to-peer overlay networks," Technical Report MSR-TR-2002-82, 2002.
[18] S. Ratnasamy, M. Handley, and R. Karp, "Topologically-aware overlay construction and server selection," in Proc. INFOCOM, 2002.
[19] M. Castro, P. Druschel, Y. C. Hu, and A. Rowstron, "Topology-aware routing in structured peer-to-peer overlay networks," in Proc. of FuDiCo 2002: International Workshop on Future Directions in Distributed Computing, Italy, June 2002.
[20] M. Castro, P.Druschel, Y. C. Hu, and A. Rowstron, "Exploiting network proximity in peer-to-peer overlay networks," Technical Report MSR-TR-2002-82, 2002.
[21] K. W. Ross and D. Rubenstein, "Tutorial on P2P systems," presented at INFOCOM 2003.

박 찬 모

1995 조선대학교 컴퓨터공학과(학사)
1997 조선대학교 컴퓨터공학과(석사)
2002 조선대학교 컴퓨터공학과(박사)
2002. 9~현재 광주과학기술원 Post Doc.
관심분야 : Peer-to-peer 공유, Overlay Multicast, 분산객체
E-mail : cmpark@netmedia.kjist.ac.kr

김 종 원



1987 서울대학교 제어계측공학과(학사)
1989 서울대학교 제어계측공학과(석사)
1994 서울대학교 제어계측공학과(박사)
1994. 3~1999. 6 공주대학교 전자공학과 조교수
1997. 7~2001. 6 미국Los Angeles, CA소재 University of Southern California, EE-Systems Dept. 방문연구 및 연구 조교수
2001.9~현재 광주과학기술원 정보통신공학과 부교수
200. 12~현재 Advanced Network Forum(ANF)산하 Application Technology Area 의장 및 Grid Forun Korea 산하 Access Grid WG 의장
관심분야 : 유무선 IP 네트워크상의 일체화된 멀티미디어의 견실하고 유연한 전달
E-mail : jongwon@netmedia.kjist.ac.kr