

DHT 기반 P2P 네트워크에서 Rich Query 매커니즘

손영성^{*} · 김희정^{*} · 김경석^{**}

^{*}한국전자통신연구원 · ^{**}부산대학교

Rich Query Mechanism in DHT based P2P Network

Young-Sung Son^{*} · Hee-Jeong Kim^{*} · Gyung-Seok Kim^{**}

^{*}Electronics and Telecommunication Research Institute · ^{**}Pusan National University

E-mail : ysson@etri.re.kr, hecjkim@etri.re.kr, gimsgs@asadal.cs.pusan.ac.kr

요 약

인터넷 상의 컴퓨팅 자원을 오버레이 네트워크를 구성하여 새로운 컴퓨팅 인프라를 구성하려는 연구가 분산 해쉬 테이블 (Distributed Hash Table) 방식의 Peer-to-Peer(P2P) 네트워크로 실현되고 있다. DHT 방식의 P2P 네트워크는 자원의 복제 및 공유하여 컴퓨팅 시스템 전반에 걸친 신뢰성과 결함감내 능력을 향상시키고 Exactly Matching Query에 장점이 있는 반면에 다중키 질의, 계층 질의 및 구간 질의와 같은 복합 질의(Rich Query) 지원에 문제점을 드러낸다. 이 논문에서는 DHT 기반의 P2P 네트워크에서 Rich Query 를 지원하기 위한 방법을 소개한다.

키워드

P2P Network, Distributed Hash Table, Rich Query

I. 서 론

인터넷의 보급에 따라 인터넷에 연결되어 있는 대다수의 단말인 PC 의 성능이 급속하게 향상되고, 인터넷에 연결되는 네트워크 속도와 대역폭이 매우 큰 반면에 이를 효율적으로 이용할 수 있는 기술은 부족하다. 이러한 현실에서 분산컴퓨팅 기술에서 발전된 Peer-to-Peer (P2P) 기술은 저비용으로 대규모 컴퓨팅이 가능한 분산 컴퓨팅 구조를 제공할 것으로 보인다. P2P 기술은 불특정 다수가 참여하는 대규모의 분산시스템의 핵심 기술로 정보검색과 정보전송, 대규모 연산 응용 등의 분야에서 성능향상과 함께 컴퓨팅 시스템 전반적에 걸친 신뢰성과 결함 감내 능력을 향상시키고 있다. 기존의 P2P 네트워크 관련 연구는 대규모 불특정 다수의 자원을 효과적으로 관리하기 위한 방법으로 분산해쉬테이블 (DIstributed Hash Table) 방식을 선택하였다. 이 방식은 자원 정보를 해쉬함수를 이용하여 임의적인 분산을 지원하였고 이를 통해서 네트워크의 부하 상황에 따른 전체 네트워크 붕괴의 위험을 해결하고 저비용의 관리 부담으로 순수 분산 환경에서 대규모의 자

원 관리를 하기에는 매우 적합하여 전체적인 시스템 효율을 높일 수 있었다. 그러나, 실제 DHT 방식의 P2P 네트워크 기술로는 사용자가 요구하는 수준의 복합 질의 (Rich Query) 를 지원할 수 없는 문제점을 보인다.

이에 본 논문에서는 기존의 DHT 기반의 P2P 네트워크 기술에서 복합 질의를 지원할 수 있는 방안을 제안한다.

II. 관련연구

2.1 Magic Square [1]

Magic Square Protocol은 시스템에 참여하는 불특정 다수의 노드들의 접속과 탈락에 영향을 덜 받는 P2P 메시지 라우팅을 지원하기 위해서 고안되었다. 각 노드는 해쉬 함수를 이용해서 m비트의 자신의 노드 아이디를 소유한다. 고정 ip 를 가진 노드는 ip 를 이용해서 노드 아이디를 만들고 가변 ip 를 가진 노드는 사용자 지정 정보를 이용해서 노드 아이디를 생성한다. 각 노드는 두 종류의 연결 테이블을 이용해 피어를 유지한다.

두 테이블은 각 노드가 시스템에 접속시에 네트워크에서 임의의 노드를 선택하는 전역 테이블(global table)과 자신의 노드 아이디를 중심으로 순차적으로 연결성을 갖는 라우팅 테이블(routing table)이다. 전체 시스템 구성은 그림 1과 같다. 지역 테이블은 양방향 스킵 리스트(bi-directed skip list)를 구성하기 위해서 사용된다. 스킵 리스트[2]는 검색 효율을 위해서 자동 조정(self-balanced) 기능을 가진 자료 구조이다. 각 노드는 자신의 네트워크 요구 처리 능력에 따라서 지역 테이블의 크기를 정한다. 그림 1은 스킵 리스트로 지역 테이블을 구성한 경우를 나타낸다. 노드 A, E, J는 네트워크 대역폭과 네트워크 처리 능력이 뛰어나고 노드 G의 경우는 전화 모뎀으로 접속되었다고 해석할 수 있다.

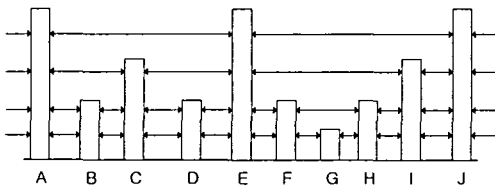


그림 1. Magic Square Routing 방식에 따른 노드 연결성 예제

2.2 Chord [3]

Chord 시스템은 P2P 응용을 위한 분산처리 자원탐색 프로토콜로, 분산 탐색을 지원하며 해쉬 함수를 이용하여 데이터의 삽입과 탐색을 수행한다. Chord는 2^k 크기의 원형 식별자 공간을 사용하며 각 노드는 IP주소를 SHA-1과 같은 해쉬 함수로 해쉬하여 nodeID를 구한 다음 원형 식별자 공간의 해당 nodeID에 위치한다. 데이터의 위치 정보는 (key, value) 쌍으로 표현되며 데이터가 저장될 노드의 위치는 key를 해쉬한 값에 의해 정해진다. 원형 식별자 공간에서 각 노드는 successor, predecessor의 정보를 유지하여 링을 형성하고 노드가 fail되었을 때 시스템을 복원하기 위해 successor list의 정보를 유지한다.

Chord에서 key를 해쉬한 값을 식별자 공간에 대응시킬 때 원형 식별자 공간에서 해쉬된 키 값과 같은 nodeID를 가지는 노드에 저장하거나 같은 nodeID를 가진 노드에 저장하거나, 같은 nodeID를 가진 노드가 없을 때는 바로 뒤의 노드에 저장한다. 이 노드를 successor 노드라 부른다. 각 노드는 전체 네트워크 상의 노드에 대한 정보를 분산된 동적 환경에서 효과적으로 만족시키기 위해 finger table을 유지한다. Finger table은 데이터를 삽입하거나 데이터를 관리하는 노드를 찾기 위해 lookup 메시지를 해당 노드에게 전달하기 위해 이용된다.

III. 문제점

다음 그림3은 DHT기반의 P2P네트워크에서 객체 저장 및 객체 질의에 대한 일반적인 방법을 설명한다. 각 피어는 전체 시스템의 유일한 해쉬 함수($h(x)$)를 가진다. 이 해쉬함수는 자신의 대표 id를 만들기 위해서 사용되고 객체 저장, 검색시에도 사용된다. 일반적인 P2P 네트워크에서 객체 저장 방법은 그림 2와 같다. 파일을 저장하기 위해서는 피어 A는 저장할 객체(a)의 특성(객체 이름, 크기, 생성날짜, 그의 정보)를 해쉬함수($h(x)$)를 이용해서 저장위치키(destination key)를 결정한다. 이 저장위치키를 이용해서 이 값을 담당하는 피어 Z를 찾아낸다. 그리고 객체(a)를 피어 Z에 저장한다. 노드B에서 해당 객체(a)를 찾기 위해서는 해쉬함수($h(x)$)를 이용해 알아낸 저장위치키를 이용해서 피어 Z를 찾아서 객체(a)를 얻는다. 이 경우에 객체(a)의 정보를 정확하게 알아야만 찾을 수 있다. 따라서, 사용자는 “최종보고서 or 중간보고서” 같은 다중키 질의 또는 “2003년 in 결과보고서”와 같은 계층 질의를 할 수 없다.

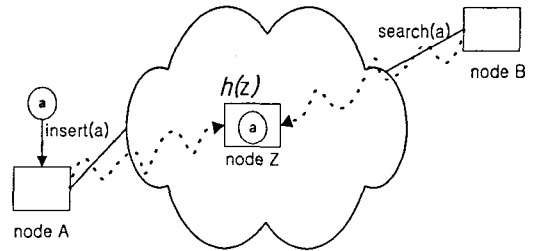


그림 2. DHT 기반 P2P 네트워크에서의 객체 저장 및 질의 방법

IV. Rich Query

DHT 기반의 P2P 네트워크에서 지원하기 어려운 복합 질의(Rich Query)는 다중키 질의(Mult-attribute query), 계층 질의(Hierarchy query)와 구간 질의(Range Query)의 세가지로 크게 구분된다.

4.1 다중키 질의

다중키 질의는 하나의 객체가 다수의 질의키를 가지는 것으로 사용자가 다수의 키를 질의하는 것을 의미한다. 예를 들어 “중간보고서 또는 결과보고서”와 같이 합집합을 요청하는 질의를 의미한다. 키워드 $K_1, K_2, K_3, \dots, K_n$ 로 이루어진 다중키 질의를 다음과 같이 표현한다.

$$MQ(K_1, K_2, K_3, \dots, K_n)$$

4.2 계층 질의

계층 질의는 하나의 질의키가 다른 질의키에 종속되는 것을 의미한다. 예를 들어 "2003년 결과보고서"는 "결과보고서"는 단독으로 의미가 있지만 "2003년"은 "결과보고서"키워드 없이는 무의미하다. 따라서 "2003년"키워드는 "결과보고서" 키워드에 종속되어 있다.

$$HQ("결과보고서" \text{ with } "2003년")$$

4.3 구간 질의

구간 질의는 하나의 객체의 값이 숫자나 행정 구역과 같이 영역을 표현할 수 있어 사용자가 이 영역의 일부 구간을 질의하는 것을 의미한다. 예를 들어 "2000년에서 2005년 동안 결과보고서"는 보고서가 작성된 연도 구간을 질의한다.

$$RQ("결과보고서" \text{ with } R(\text{year}, 2000, 2005))$$

V. Rich Query 지원 방법

5.1 다중키 질의 지원 방법

다중키 질의에 포함되는 모든 키별로 객체를 수집하여 이를 교집합 또는 합집합을 구성하여 질의 결과를 얻을 수 있다. 그림 3은 다중키 질의 방법을 나타낸다. 각 키별로 질의를 하여 그 결과를 후처리하여 최종 질의 결과를 만든다.

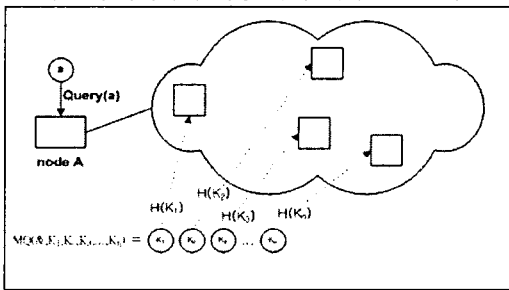


그림 3. 다중키 질의 방법

이를 표현식으로 나타내면 다음과 같다.

$$MQ(\&K_1, K_2, K_3, \dots, K_n) = \{H(K_1) \& H(K_2) \& H(K_3) \dots \& H(K_n)\}$$

$$MQ(U, K_1, K_2, K_3, \dots, K_n) = \{H(K_1) | H(K_2) | H(K_3) \dots | H(K_n)\}$$

5.2 계층 질의 지원 방법

계층 질의의 경우는 주 키워드로 검색하여 그 결과에서 부 키워드로 검색해야 한다. 계층 질의를 다중키 질의로 처리하면 부 키워드 검색으로 나온 결과 수가 매우 클 가능성이 있다. 따라서 계층 질의를 지원하기 위해서는 객체를 저장시에 주 키워드로 저장하고 주 키워드의 해쉬주소와 부 키워드의 해쉬 주소를 조합하여 다시 저장해야 한다.

$$HQ(K_1 \text{ with } K_2) = Q(Q(K_2) \wedge Q(K_1))$$

5.3 구간 질의 지원 방법

DHT 기반의 P2P 네트워크에서 구간 질의는 사실상 지원하기 어렵다. 구간 질의 지원을 위해서

객체 저장 및 질의에 특수한 온톨로지 기반 키워드를 사용하고 객체가 저장되는 주소를 파악하여 이를 포함하는 구간을 각 키워드 별로 특징에 맞게 확대하여 이 구간을 영역별로 구분하여 관리한다. 그림 4에서는 "보고서" 객체의 주소가 F와 G 사이이다. 이 객체가 "year" 온톨로지 타입을 가지고 있기 때문에 "보고서 with X" 질의는 노드 E에서부터 노드 J까지 모두 검색해야 한다.

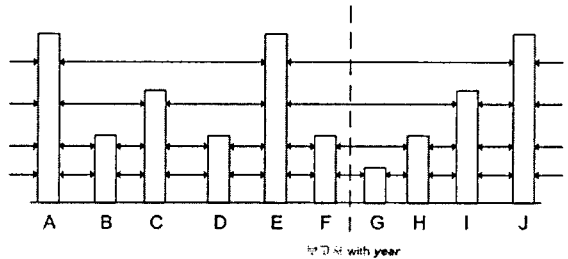


그림 4. 구간 질의 방법

VI. 결 론

이 논문에서는 DHT 기반의 P2P 네트워크에서 지원하기 어려운 Rich Query를 지원하기 위한 방법을 소개한다. 기존의 MagicSquare P2P Network의 특징을 살려서 다중키 질의, 계층 질의, 구간 질의를 지원하기 위한 방법을 제시하였다.

참고문헌

[1] Sun-Mi Park, "Magic Square: Scalable peer-to-peer lookup protocol considering peer's characteristics", IASTED CIC, 2003.
 [2] William Pugh, "Skip Lists: a probabilistic alternative to balanced Trees", communications of the ACM, Vol.33 No.6 pp.668-676. June 1990
 [3] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek and H. Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications," In Proceedings of SIGCOMM (August 2001)
 [4] K. Shin, S. Lee, G.. Lim, H. Yoon, and J. S. Ma, "Grapes: Topology-based Hierarchical Virtual Network for Peer-to-peer Lookup Services," In Proceedings of the International Conference on Parallel Processing Workshops (ICPPW' 02), 2002.