MagicSquare를 이용한 애플리케이션 레벨 멀티캐스트

김강범° 한동윤, 손영성† 김경석‡
부산대학교 컴퓨터공학과, 한국 전자통신 연구소†, 부산대학교 정보컴퓨터공학부‡
{gbkim°, dyhan, ysson, gimgs}@asadal.cs.pusan.ac.kr

Application-level Multicast Using Magic Square

Gang-beom Kim° ,Dong-Yun Han, Young-Sung Son†, Kyongsok-Kim‡
Dept of Computer Science and Engineering, Electronics and Telecommunications Research Institute,
Devision of Computer Science and Engineering. Pusan National University

요 약

IP 멀티캐스트 기술은 지난 10여년동안 꾸준히 발전해 왔지만 라우터의 편재형 배치(Ubiquitous deployment)가 되지 않음으로 인해 네트워크 라우터가 패킷을 복사, 전달함으로서 종단 호스트와 네트워크 라우터 간에 부가적인 제어와 라우팅 프로토콜이 필요하게 되는 단점이 있다. 이에 대한 대안으로 현재 오버레이 네트워크를 이용한 애플리케이션 레벨 멀티캐스트가 많이 연구되고 있다. 본 논문에서는 P2P 오버레이 네트워크를 이용해 확장성(scalability)이 있고 참여지연(Join delay)이 적으면서 효율적인 트리를 구성할수 있는 애플리케이션 레벨 멀티캐스트를 제안한다.

1. 서 론

IP 멀티캐스트가 라우터의 배치 문제로 인해서 그 대안으로 애플리케이션 레벨 멀티캐스트가 제시되었다. 애플리케이션 레벨 멀티캐스트 방식은 크게 메시(mesh) 기반 방식과 트리(tree) 기반 방식으로 나눌 수 있다. 메시 기반 방식은 최적화된 소스 기반 데이터 분배 트리를 만들 수 있으나 컨트롤 오버헤드가 크기 때문에 그룹 사이즈가 조그만 사이즈 애플리케이션으로 제한된다. 그에 반해 트리 기반 방식은 비교적 컨트롤 오버헤드가 적기 때문에 확장성이 있다. 그러나 데이터 분배 트리가 최적화가 되지 않는다는 문제점이 있다. 트리 기반 방식에서는 자신과 가장 가까운 노드를 선택하기 위해서 한 단계씩 내려가면서 트리 내의 노드들과의 거리를 측정함으로서 그룹에 참여하는 시간이 길어지게 된다. 또한 노드에 제한 차수가 있기 때문에 자신과 가장 가까운 노드를 실제로 선택하지 못하므로 트리의 효율성이 떨어진다. 참여하는데 걸리는 시간이 클 경우에 부모노드가 갑작스럽게 고장(fail)이 발생하면 다시 그룹에 참여하는 시간이 길기 때문에 스트리밍 같은 경우에는 끊어지는 시간이 길어지게 된다. 본 논문에서는 Magic square[1] 기반으로 확장성이 있으며 그룹에 참여하는 시간이 빠르고 노드의 팬아웃(fanout) 제한 없이 노드간의 거리를 고려한 애플리케이션 레벨 멀티캐스트 트리를 제시한다. 본 논문의 구성은 다음과 같다. 2장에서는 관련연구에 대해서 살펴본다. 3장에서는 트리를 구성하는데 사용될 Magic Square에 대해서 살펴본다. 4장에서는 그룹 생성과 그룹의 멤버들을 관리하는 방법에 대해서 살펴본다. 마지막으로 5장에서는 결론을 내린다.

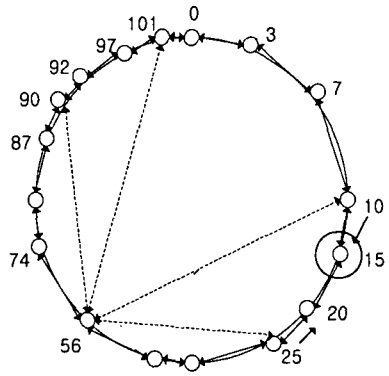
2. 관련연구

p2p 기반 애플리케이션 레벨 멀티캐스트에는 Pastry [2]를 이용한 Scribe[3]와 Tapestry[4]를 이용한 Bayeux[5], 그리고 CAN[6]을 이용한 CAN Multicast[7]가 있다. CAN Multicast는 큰 그룹 사이즈로 구성이 가능한 방법들이 단일 소스(single source) 모델로 제한되어 있다는 문제를 해결하여 다중 소스(multi source) 모델로 사용이 가능하게 했다. CAN multicast는 트리를 구성하지 않는다. 대신에 CAN에 의해서 유지되는 라우팅 테이블들을 사용하여 CAN 오버레이 네트워크 내에 있는 모든 노드에게 메시지를 플루딩(flooding)하는 방식이다. Scribe는 Pastry 네트워크 위에 그룹당 하나의 멀티캐스트 트리를 구성한다. Pastry는 latency에 기반을 두고 루트에서 각각의 그룹멤버에 대해서 경로를 최적화하는 것을 보장한다. Bayeux는 Tapestry 네트워크 위에 그룹당 하나의 멀티캐스트 트리를 구성하는점에서 Scribe와 비슷하나 트리 구성방식이 다르다. 루트가 새로운 멤버를 기록하고 거기에 대한 응답으로 새로운 멤버에 메시지를 보내주는 방식이다.

3. MagicSquare

기존 탐색 프로토콜(CAN,Tapestry,Pastry)에서 피어는 자신의 능력과 상관없이 같은 비율의 질의를 처리한다. 네트워크 처리능력이 낮은 피어는 네트워크 처리능력이 높은 피어에 비해 상대적으로 부하가 많이 걸리게 되고 이는 전체 시스템의 성능을 저하시킨다. Magic Square에서는 네트워크 처리능력이 높은 피어가 많은 수의 질의를 처리하게 하도록 라우팅 테이블을 구성함으로서 좀 더 빠른 응답시간을 가진다. 라우팅 테이블을 스킵리스

트(Skip List)[8]로 구성함으로써 새로운 피어의 추가 또는 삭제의 과정이 간단하기 때문에 피어의 잦은 접속/단절에 안정적이고 P2P네트워크가 자주 바뀌어도 큰 영향을 받지 않는다.



[그림 1 Magic square에서의 자원탐색]

4. 그룹생성 및 멤버관리

4.1 트리 루트

트리구조는 부모의 선택이 성능을 결정한다. 따라서 트리의 가장 상위에 있는 부모노드인 루트를 선정하는 것은 중요하다. 예를 들어서 P2P 오버레이 네트워크에 참여한 노드가 모뎀과 같은 낮은 대역폭을 사용할 경우에 이 노드가 루트가 되면 멀티캐스트 성능이 떨어질 수 있다. 기존 논문에서는 트리의 루트를 선정할 때 노드의 능력을 고려하고 있지 않다. Scribe와 Bayeux의 경우에는 그룹의 컨텐츠 이름을 해싱(hashing)해서 나오는 ID와 숫자상 가장 가까운 노드를 루트로 하거나 처음 그룹에 들어온 노드를 트리의 루트로 선택한다.

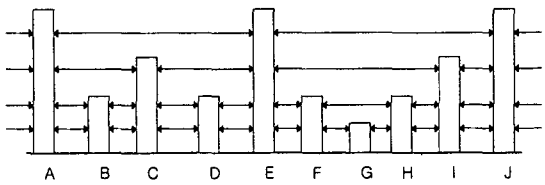
Magic square에서 레벨(level)이 큰 노드를 루트로 하는 또 다른 이유는 루트로 찾아갈 때 홉(hop)수를 줄일 수 있기 때문이다. 레벨이 큰 노드일수록 많은 노드의 정보를 포함하고 있기 때문에 접근(access) 확률이 높아진다. 즉 레벨이 높은 노드의 접근(access)빈도가 높다. 즉 트리가 구성될 때 레벨이 작은 노드보다 레벨이 큰 노드의 자식이 될 확률이 높아진다.

처음 그룹의 tree의 root를 만들때 그룹을 만들려는 노드는 자기가 가지고 있는 지역,전역 라우팅 테이블을 조사하여 가장 레벨이 큰 노드를 루트로 선택을 한다. 이렇게 할 경우에 문제점은 레벨이 작은 노드일수록 라우팅 테이블에 포함되어 있는 노드의 정보가 적기 때문에 높은 레벨을 가진 노드가 라우팅 테이블에 포함되어 있을 확률이 레벨이 높은 노드보다 떨어진다는 것이다. 그래서 선택한 루트가 레벨이 낮은 루트일 경우에는 일단 루트를 선택하고 나서 루트가 고장이 났을때를 대비하기 위해 일정 개수의 다중 루트(multiple root)를 만들게 되는데 거기서 얻어지는 다중 루트 집합에서 레벨이 가장 높은 것을 루트로 선택을 한다.

루트 뿐만이 아니라 부모노드에 팬아웃(노드에 걸리는 다른 노드의 수)을 제한할 필요가 있다. 실제 하나의 호스트가 동시에 많은 수 이상의 연결을 지원하기 위한 대역폭을 지니지 못하기 때문에 트리를 구성할 때 노드의 팬아웃을 제한해야 된다. 본 논문에서는 팬아웃을 실제로 제한하지는 않는다. 그 이유는 레벨이 높은 노드에 많은 수의 자식이 생기므로 레벨이 낮은 노드에는 적은 수의 자식이 생기는데 노드의 능력에 따라 적절히 팬아웃 차수가 결정이 되기 때문이다. 능력이 높은 노드는 많은 팬아웃 차수를 가지게 하고 능력이 낮은 노드는 적은 차수를 가지고 함으로써 합리적으로 팬아웃 차수를 제한한것과 같은 효과를 낸다.

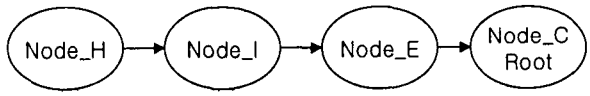
4.2 멤버 조인

노드가 새로운 그룹에 참여할 경우 일단 트리 루트로 조인(join) 메시지를 보낸다. 조인 메시지가 트리 루트로 가는 과정은 magic square의 라우팅 기법을 사용한다. Member들은 자식들과 부모의 정보를 유지한다.

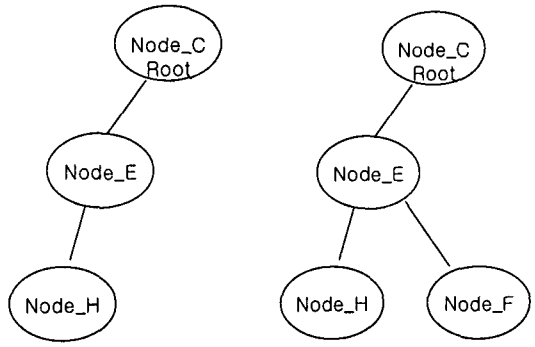


[그림 2] Magic Square 기본구조

[그림 2]과 같은 Magic Square 상태를 고려하자. 트리 루트가 노드 C이고 그룹에 노드 E가 자식으로서 들어와 있다고 하자. 그룹에 참여 하려는 노드가 H라고 할때 노드 H는 노드 I와 E를 통해 트리 루트 C로 가게된다. 라우팅 경로는 아래 [그림 3-1]과 같고 실제적인 데이터가 전송되는 멀티캐스트 트리는 아래 [그림 3-2]와 같다.



[그림3-1 라우팅 경로]

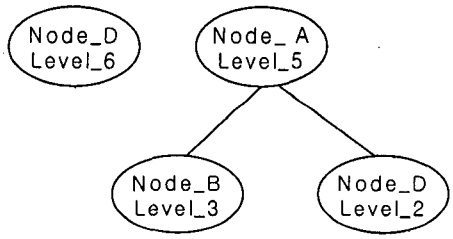


[그림3-2 노드 F 조인되기전] [그림3-3 노드 F 조인된후]

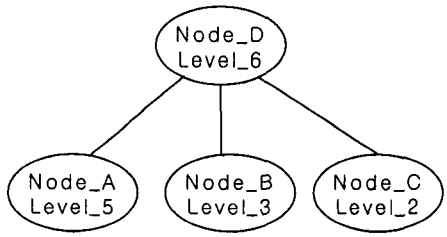
[그림 3-1]에서 노드 I 는 그룹의 멤버가 아니다. 이런 노드를 전송자(Forwarder)라고 하자. 전송자는 자신의 부모노드(루트 또는 멤버)를 자신의 부모 테이블에 유지한다. 새로운 멤버가 참여할려고 할때 전송자를 만나게 되면 조인 메시지는 루트로 전송이 되지 않고 제거되며 해당 전송자는 참여할려는 노드를 자신의 자식 테이블에 추가한다. 위의 [그림 3-2]의 상태에서 노드 F가 그룹에 참여할 경우에는 노드 E를 통해 루트로 가기 때문에 노드 E의 자식이 된다. [그림 3-3]

Magic square의 특징이 레벨이 높은 노드가 레벨이 낮은 노드보다 접근이 많이 되기 때문에 노드 F는 노드 H의 자식이 될 확률보다 노드 E의 자식이 될 확률이 많다. 노드 E와 노드 F 사이의 거리가 노드 E와 노드 H 사이의 거리보다 가깝다. 즉 Magic square 특징을 이용해서 거리를 고려한 트리를 구성할 수 있다.

만약 어떤 노드가 부모를 선택하여 그 노드의 자식이 되고자 한다면 부모노드보다 자식노드가 레벨이 높을 경우에는 부모와 자식을 바꾼다.



[그림 4-1] 노드가 조인하기 전



[그림 4-2] 노드가 조인된 후

[그림 4-1]에서 노드 D가 노드 A에 자식으로 들어갈려고 하는데 노드 A가 노드 D보다 레벨이 낮다. [그림 4-2]에서 노드 D가 노드 A와 교체되고 노드 A는 노드 D의 자식이 된다. 노드 A는 자신이 가지고 있는 자식 테이블의 정보와 부모 정보를 노드 D에 넘겨준다. 이렇게 함으로서 쉽게 부모와 자식을 교체할수 있다.

4.3 노드의 그룹탈퇴

4.3.1. 그룹 내의 멤버들이 정상적으로 종료된 경우

자식이 없는 노드가 떠날 경우에는 부모노드에게 자신이 나간다는 사실을 알려준다. 자식이 있는 노드가 I 떠날 경우에는 부모노드에게 자식의 정보를 넘겨준다.

4.3.2. 그룹 내의 멤버들이 비정상적으로 종료되었을 경우(갑작스런 종료)를 대비해서 각각의 노드들은 자신의 부모와 자식들이 살아있는지 정기적으로 체크 메시지를 보낸다. 자식이 없는 노드일 경우에 일정 시간동안 응답이 없으면 비정상적으로 종료된 노드의 부모는 자식 테이블에서 노드에 대한 정보를 삭제한다. 자식이 있는 노드일 경우에는 즉 부모가 고장이 난 경우이기 때문에 부모에 대한 자식들은 그룹에 다시 조인을 한다.

4.4 멀티캐스트 트리 루트의 고장

일단 트리를 루트를 만들때 트리의 루트가 고장이 났을 때를 대비하기 위해서 인접한 일정한 개수의 루트를 만든다. 트리의 루트가 고장되었을때는 루트의 바로 이웃한 노드가 새로운 트리 루트를 찾는다.

5. 결론

본 논문에서는 Magic Square P2P 프로토콜에 기반을 둔 애플리케이션 레벨 멀티캐스트를 제안하였다. Magic square의 라우팅 경로를 이용해 트리를 구성하고 각 노드에 부여된 레벨을 이용하여 노드간의 거리를 고려한다. 실제 팬아웃 제한 없이도 레벨에 따라 적절히 팬아웃 차수가 결정이 된다.

참고문헌

[1] 박선미,정일동,손영성,김경석, MagicSquare: 노드의 능력을 고려한 자원 프로토콜. 2003 봄 학술발표논문집(C) 제30권 1호 pp.163~165 한국정보과학회 (2003/04)

[2] ROWSTRON, A., AND DRUSCHEL, P. Pastry: Scalable, distributed object location and routing for large-s cale peer-to-peer systems. In Proceedings of the 18th IFIP /ACM International Conference on Distributed Systems Platforms (Middleware 2001) (Nov. 2001)

[3] Castro, P. Druschel, A.-M. Kermarrec, and A.Rowstron Scribe: A large-scale and decentralized application-level multicast infrastructure

[4] Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry: An Infrastructure for Fault-Resilient Wide-area Location and Routing. Technical Report UCB//CSD-01-1141,U. C. Berkeley, April 2001.

[5] S. Q. Zhuang, B. Y. Zhao, A. D. Joseph, R. H. Katz, and J. Kubiatowicz. Bayeux: An Architecture for Scalable and Fault-tolerant Wide-Area Data Dissemination. In Proceedings of the Eleventh International Workshop on Network and Operating System Support for Digital Audio and Video, June 2001.

[6] Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, "A Scalable Content-Addressable Network," SIGCOMM, 2001.

[7] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Application-level Multicast using Content-Addressable Networks.

[8] William Pugh. "Skip List : a probabilistic alternative to balanced Trees" , communication of ACM 33권 6호