2001 년 한국정보처리학회 춘계 학술발표논문집 제 8 권 제 1 호

Ring 을 기반으로 하는 클러스터 순수 Peer-to-Peer 네트웍 설계

전상현, 구태완, 정연진, 이광모
한림대학교 컴퓨터공학과

e-mail : shcheon@cie.hallym.ac.kr

A Ring-based Cluster Pure Peer-to-Peer Network Design

Sang-Hyun Jeon, Tae-Wan Gu, Yeon-Jin Jeong, Kwang-Mo Lee
Dept. of Computer Engineering, Hallym University

요 약

인터넷의 활용이 급속도로 확산되어지고 사용환경도 점점 고급화가 되어가고 있다. 분산환경에는 네트웍에 연결된 각 Peer 들의 유휴자원을 이용하여 대규모의 데이터를 처리하기 위한 연구와 기존의 클라이언트-서버 환경에서의 부하를 각 Peer 들에게 분산시키는 연구가 활발하게 진행되고 있다. 이 연구에서는 서버의 지원이 없는 순수 Peer-to-Peer 환경에서 현재의 Peer 들의 네트웍 구성을 살펴보고 효율적으로 중복된 메시지를 제어하기 위한 Ring-based 클러스터를 통한 네트웍 구성 환경과 클러스터간의 메시지 전파와 대역폭에 따른 클러스터 기법을 제안한다.

1. 서론

분산 시스템의 주요 디자인 접근방법은 클라이언트-서버 모델(이하 C/S)과 Peer-to-Peer(이하 P2P)모델이다. 인터넷 사용자가 급속도로 증가하면서 파일 전송 서비스를 하기 위한 서버나 대규모의 채팅을 지원하기위한 솔루션은 고성능의 서버와 큰 저장매체를 구축해야만 한다. 또한, 기존의 메일 시스템은 여러 서버를 경유해야 하므로 보안문제와 지연 전송 문제가 대두되었다. 위에서 제시한 하나의 대안으로 대두되는 개념이 P2P 모델이다. 초고속 통신망과 사용자의 컴퓨터가 점점 고성능화가 되므로 P2P 모델에서는 이들 사용자의 컴퓨터(피어)를 활용한다. 각각의 피어들은 네트웍 서비스를 유지하기 위한 같은 책임을 공유한다[1]. 구조적으로는 서버의 기능과 클라이언트를 결합한 형태로 파일 전송이나 채팅 솔루션도 각 피어들간에 전송이 이루어지므로 서버가 없거나 기존의 C/S 모델에 비해 서버의 작업량은 적어지고, 직접 전송하므로 빠르다.

이 논문의 구성은 다음과 같다. 2 장에서는 기반 모델에 대해 설명하며 3 장에서는 기반 모델에서의 순수 P2P 의 네트웍 구성에 관한 기존 연구를 살펴보고, 4 장에서는 링기반 클러스터 구조에 대해 설명하고, 효율적인 피어의 관리 기법에 대해 제안하고, 5 장에서는 향후 연구 과제를 살펴본다.

2. P2P 기반 모델

P2P 기반 모델은 피어들 간의 연결을 설정에 따라 다음과 같이 세가지로 분류한다.

1)하이브리드 P2P

중계자인 브로커를 두어 피어들의 상호작용을 촉진하는 역할을 하며, 냅스터와 같은 방식으로 중앙 서버가 있는 경우를 말한다. 그림 1 과 같이 피어들 간의 연결과 빠른 검색을 지원하기 위한 피어들의 목록을 유지하는 방식이다[2][3]. 그러나, 기존의 C/S 모델에 비해 파일 전송과 같은 오버헤드가 많은 일은 피어들 간에 이루어지므로 효율적이다.

오류! 연결이 잘못되었습니다.

2)순수 P2P

하이브리드 방식과는 다르게 브로커가 없는 방식으로 그림 2 와 같이 피어들의 메시지 패싱에 의해 파일을 공유하는 방식이다.

오류! 연결이 잘못되었습니다.

3)사이클 공유

그림 3 과 같이 마스터로부터 피어는 데이터를 전송 받아 피어가 유휴 상태일 경우에 데이터를 처리 마스터에 돌려 준다.

오류! 연결이 잘못되었습니다.

3. 순수 P2P 네트웍 구성

1)격자 구조.

격자 프로토콜은 정규 위상을 가진 네트웍을 만들기 위하여 디자인 되었고, 가장 효율적인 멀티캐스트 메시지를 처리하고, 사용하는 네트웍 구조는 직사각형 격자이거나 메쉬 형태를 가진다. [6]

오류! 연결이 잘못되었습니다. 2)계층 구조

네트웍은 다양한 피어들의 컴퓨터로 연결되어 있고, 그들이 사용하는 환경도 다양하다. 모뎀을 사용하는 피어는 대부분의 전송시간을 점유할 가능성이 높기 때문에 더 빠른 네트웍을 사용하는 피어의 하부 구조로 두었다. 이 구조에서는 이러한 제한을 두어 좀더 효율적인 구성을 제시하였다[7][8].

오류! 연결이 잘못되었습니다. 3)클러스터 구조.

위에서 제시한 격자구조를 개선한 구조로서 각 클러스터는 노드들의 집합인 "co-nodes"를 가지며, co-nodes 의 세부구조는 작은 격자구조를 따른다[9].

오류! 연결이 잘못되었습니다.

4. 링기반 클러스터 구조.

격자구조는 모든 네트웍을 정규구조를 따름으로 각 노드는 4 개의 다른 노드와 연결함으로써 환형구조를 제거하기에는 효율적이다. 그러나, 연결을 하는 오버헤드가 크고, 유지 보수가 어렵다는 단점이 있다. 예로 들면, 100 개의 노드가 격자구조를 따르고 있다고 가정할 경우, 그 중 10 개의 노드가 연결을 해제했다고 하면, 10 개의 홀이 생기고, 그 홀은 다시 채워질 수가 없다. 또한 이 구조에서는 네트웍을 사용하는 노드들의 대역폭을 고려하지 않았다. 그러한 문제점은 모든 대역폭이 한 노드에서 걸림으로써 전체적인 성능의 저하를 가져온다[9]. 또한, 계층 구조에서는 4 개의 노드 타입을 정하며 타입은 다음과 같다.

[A]: Mini-Node [B]: Node

[C]: Super-Node [D]:Mega-Node

각 노드의 유형은 미리 결정되며 각각의 노드에는 연결에 관한 제한이 가해진다. [A]노드는 연결을 가질 수 없고 [B]노드는 5 개까지 가질 수 있다. 이런 규칙에 의해 노드들이 연결되므로 [D]노드는 [C], [B], [A] 노드들의 집합을 갖는다. 그러므로 상위 노드일수록 많은 연결 상태를 가지고 있으며 부하가 많이 걸린다. P2P 에서의 피어(노드)는 사용자가 느끼지 못할 정도의 부하만을 허용해야 함에도 불구하고 계층구조는 상위노드에게 의존성이 강하다. 이런 문제는 피어의 상태는 불안정하다는 현실에 위배된다고 볼 수 있다. 즉 상위 노드의 시스템이 다운되었을 경우 하부노드는 다른 상위 노드에 연결이 되므로 그 노드는 역시 과부하 상태로 될 수 있다.

클러스터구조는 격자구조를 기반으로 하여 클러스터 내부에서는 환형을 제거하였고, 클러스터 ID 를 두어 다른 클러스터와의 메시지를 전달한다. 그러나, 클

러스터 ID 의 유일성을 보장하지 못했고, 내부에서의 환형은 제거하였으나 클러스터 사이의 환형은 그대로 남아있다.

이에 본 논문에서는 링을 기반으로 하는 클러스터를 제안한다. 제안 모델은 그림과 같다.

오류! 연결이 잘못되었습니다. 링기반 클러스터 모델에서는 계층적인 구조에서의 노드 구별을 이용하여 그림 7 에서의 색이 없는 노드는 [A]노드와 같고, 나머지는 구별하지 않는다. 즉, 오버헤드를 많이 유발하는 Hog 노드만을 제외하였고, Election 알고리즘을 사용하여 노드 중 가장 신뢰성이 높은 노드를 코디네이터로 선출한다[10][11][12].

[GA]: 계층구조의 [A]

[GB]: 계층구조의 [B][C][D]

[GA]에 속한 노드는 자신의 하부구조를 가질 수 없고 오직 [GB] 의 하부구조로 속할 수 있다.

[GB]에 속한 노드는 5 개 이하의 노드를 가질 수 있다. 이런 제약은 [GB]에 속한 노드의 부하를 줄이기 위한 제약이다. 클러스터도 [GB]에 속한 노드에 제약을 주어야 한다. 여기서는 7 개 이하로 정하였다. 링을 기반으로 하기 때문에 클러스터의 노드들을 순회하기 위한 시간을 적게 하기 위함이다. 노드들이 순회는 [GB]에 속한 노드가 시스템의 실패했을 경우에 링을 재구성하기 위해서 사용한다. [GA]의 노드는 [GB]의 노드만을 기억한다.

코디네이터는 클러스터 1 의 노드들이 노드 A 에게 클러스터 ID 를 요청하면 생성하여 되돌려 준다. 제안한 모델에서의 클러스터 ID 는 IP 주소를 사용한 java.util.Hashtable 의 hashCode()를 이용할 수도 있다. 각 노드는 자신의 정적인 변수를 두어 Query 할 경우에 유일한 메시지의 ID 를 생성한다. 클러스터 내부는 링형 구조이므로 환형이 발생하지 않는다. 각 클러스터에서 각 노드는 코디네이터에게 클러스터 ID 를 구해서 자신의 메시지 ID 를 추가한 Query 를 수행함으로써 클러스터사이의 환형을 제거한다. P2P 환경에서는 노드들의 신뢰성을 기반으로 코디네이터를 선정하여야 한다. 이 모델에서의 가장 크게 발생할 수 있는 문제는 코디네이터의 변경 때문에 발생한다.

현재의 건델라는 0 바이트~15 바이트를 메시지 ID 로 사용하고 있다[13]. 이것은 윈도우의 GUID 로 메시지를 다시 처리하지 않도록 하기 위해 사용된다. 건델라의 프로토콜의 해더구조를 본 논문에서 제시한 구조에 맞도록 변경한 것이 표 1 이다.

바이트	의미	설명
0~3	클러스터 ID	코디네이터의 Hashcode 생성 값
4~7	메시지 ID	자신이 생성한 메시지의 ID
8	패킷 종류	추가 패킷종류 값 패킷 종류 0x20 클러스터 ID 요구 0x21 클러스터 ID 응답

표 1 헤더 구조

클러스터 ID 와 메시지 ID 를 더하여 유일성을 갖는 GUID 와 같은 역할을 한다. 건텔라 프로토콜에서의 TTL(Time To Live)은 메시지가 전달될 때마다 하나씩 감소하여 0 이 되면 전달하지 않고 소멸된다. 이 모델에서의 TTL 은 클러스터의 내부에서는 전혀 사용되어지지 않고 클러스터들의 사이에서만 하나씩 감소한다. 처리한 메시지의 클러스터 ID 와 메시지 ID 는 캐시를 수행하여 중복된 처리를 제거한다.

5. 결론 및 향후 연구 과제

P2P 환경은 앞으로 전반적이고 세계적인 네트웍을 형성할 수 있는 기반 기술로 자리잡은 것이다. 이 논문에서는 글로벌 네트웍을 위한 순수 P2P 환경을 위한 피어들의 링을 기반으로 하는 클러스터기법에 대해 논의하였다.

향후 과제로는 피어가 클러스터에 조인될 때 접속이 가장 빠른 클러스터를 찾는 기법을 이용하여 피어들을 동적으로 재구성될 수 있도록 설계되어야 한다.

참고문헌

[1] John Huebner, Brad A, Myers,"Easily Programmable Shared Object For Peer-to-Peer Distributed Applicatins".

[2] 이재규. "파일 공유를 넘어선 P2P 의 실체 분석",10 월호. 마이크로 소프트, 2000

[3] 박종의. "인터넷 혁명 P2P",10 월호. 프로그램 세계, 2000

[4] http://www.endtech.com, "Introducing Peer-to-Peer".

[5] Bob Nighten, "Peer-to-Peer Computing", August 24,2000,Intel Developer's Forum.

[6] Ben Houston, "The Grid P2P Topology Protocol.

[7] Joseph Wasson. "Proposal For the New Gnutellesque Architecture.

[8] Sebastien Lamda, " Monaco Proposal, 2000/4/14

[9] Philip Ganchev, "Clusters – a Proposed Topology for a Peer-to-Peer Network.

[10] Chang, E.G. and Robert,R. "An improved Algorithm for decentralized extrama-finding in circular configurations of processors. Comm. ACM, Vol.22, No.5, pp. 281-3.

[11] George Coulouris, Jean Dollimore, Tim Kindeberg "Distributed Systems Concepts and Design", pp432-4,3rd Edition, Addison-Wesley.

[12] Nancy A.Lynch "Distributed Algorithm", pp25-50, Morgan kaufmann.

[13] The Gnutella Protocol, http://gnutella.wego.com.