클러스터 P2P 네트워크에서의 최적 슈퍼피어 개수

김 성 희[†] · 김 주 균[††] · 이 상 규[††] · 이 준 수[†††]

요 약

슈퍼피어 기반 P2P 네트워크는 전체 네트워크를 여러 개의 작은 서브 네트워크로 클러스터링하고 각 클러스터를 해당 그룹에 속한 노드들에 대한 정보를 가지고 있는 슈퍼피어라는 특정 노드가 관리하는 네트워크 모델로써 검색의 효율성과 네트워크 부하가 적다는 이점을 가지고 있다.

본 논문은 슈퍼피어기반 P2P네트워크에서 먼저 피어들의 정보검색, 새로운 노드 가입, 정보갱신 등의 동작으로 발생하는 메시지의 양을 기반으로 한 트래픽 비용을 클러스터 내의 비용과 슈퍼피어 간의 비용으로 측정하고, 이 두 비용을 바탕으로 다양한 네트워크 크기에 따라 트래픽 비용을 최소화할 수 있는 슈퍼피어의 개수를 제시한다.

키워드 : P2P 네트워크, 슈퍼피어, 클러스터, 트래픽 비용

Optimal Number of Super-peers in Clustered P2P Networks

SungHee Kim[†] · JuGyun Kim[††] · SangKyu Lee[††] · JunSoo Lee[†††]

ABSTRACT

In a super-peer based P2P network, The network is clustered and each cluster is managed by a special peer, called a super-peer which has information of all peers in its cluster. This clustered P2P model is known to have efficient information search and less traffic load.

In this paper, we first estimate the message traffic cost caused by peer's query, join and update actions within a cluster as well as between the clusters and with these values, we present the optimal number of super-peers that minimizes the traffic cost for the various size of super-peer based P2P networks.

Key Words : P2P Network, Super-peer, Cluster, Traffic Cost

1. 서 론

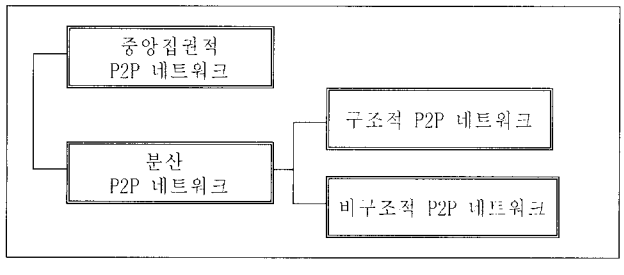
P2P(Peer-to-Peer)네트워크는 네트워크에 참여하는 노드를 피어라 지칭하고 그 노드들이 서버와 클라이언트의 역할을 동시에 담당하여 노드가 가지고 있는 자원의 일부를 공유하는 분산시스템을 말한다[1]. 기존의 클라이언트-서버 네트워크는 모든 서비스를 서버가 처리하는 형태이므로 서버의 성능이 떨어지거나 과도한 요청으로 인해 클라이언트와 서버의 연결이 원활하지 않게 되면 전체 네트워크 서비스의 질이 떨어지게 될 수밖에 없다. 이러한 서버 집중에 의한 문제점을 개선하기 위한 대안으로 나온 것이 P2P네트워크 모델이다. P2P네트워크는 많은 피어들의 저장장치나 프로세서들을 공동으로 활용할 수 있고, 각각의 피어에 저장되어 있는 각종 미디어 컨텐츠들이나 공동 작업에 필요한 파일 등의 공유 기능도 지원할 수 있는 큰 이점을 가지고 있어 중요한 인터넷 서비스 중의 하나로 운영되고 있다[2].

초기의 P2P네트워크 모델인 냅스터(Napster)는 순수하게 분산되어 각 피어들이 독립적으로 기능을 수행하는 형태는 아니다. 오히려 중앙의 서버가 존재하여 전체 네트워크의 데이터를 관리하는 네트워크 모델이다. 그러나 기존의 클라이언트-서버 네트워크에서의 서버가 클라이언트가 요구하는 서비스와 관련한 모든 일을 수행하지만, 냅스터의 서버는 클라이언트에 해당하는 피어들의 데이터에 대한 인덱스만 보유하여 검색 요청 시 그 인덱스를 검색하고 원하는 정보를 가진 피어와 직접 연결하도록 도와주는 역할만 한다[3,4].

(그림 1)은 P2P 네트워크 모델을 구조적 특성에 따라 분류한 것이다. 현재 P2P네트워크에 대한 연구는 냅스터의 중앙집권적인 네트워크 모델을 탈피하여 분산된 네트워크에

※ 본 연구는 숙명여자대학교 2006년도 교내연구비 지원에 의해 수행되었음
† 정 회 원 : 삼성전자 정보통신총괄 통신연구소 차세대단말팀 선행단말Lab
†† 정 회 원 : 숙명여자대학교 정보과학부 컴퓨터과학전공 교수
††† 정 회 원 : 숙명여자대학교 정보과학부 컴퓨터과학 전공 조교수
논문접수 : 2006년 3월 24일, 심사완료 : 2006년 7월 13일

대한 연구로 전개되어지고 있다. 분산된 P2P네트워크는 비구조적 P2P네트워크와 구조적 P2P네트워크로 분류된다[5]. 그누텔라(Gnutella)[6], 프리넷(Freenet)[7]등의 비구조적 P2P 네트워크는 모든 피어들이 피라미드식으로 연결되어 무제한으로 자료를 공유할 수 있고 특정 서버 없이 서비스 제공이 가능하다는 장점을 가지고 있다. 하지만, 원하는 정보검색요청을 네트워크에 연결된 모든 피어들에게 보내기 때문에 중복 트래픽(traffic)이 과도하게 발생하고 검색의 완전성에도 문제점이 발생할 수 있다는 단점을 가지고 있다[8]. 구조적 P2P 네트워크는 이름 그대로 비구조적 P2P 네트워크의 문제점을 극복하기 위해 구조적인 특성을 부여한 것이다. 대표적으로 실제 값과, 해쉬 함수를 적용하여 형성한 키 값의 조합을 사용하여 검색을 빠르게 하는 DHT(Distributed Hash Table) 기반 P2P 네트워크가 있다[9~13]. 그러나 DHT 기반 P2P네트워크는 데이터의 키 값을 완벽하게 알지 않으면 검색의 수행이 어렵고, 네트워크의 물리적인 상황을 고려하지 않는다는 문제점이 있다.



(그림 1) 구조적 특성에 따른 P2P 네트워크 모델 분류

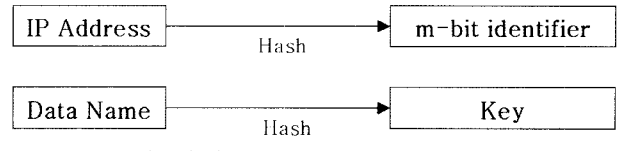
슈퍼피어(Super-peer) 기반 P2P네트워크는 또 다른 구조적인 P2P네트워크로써, 네트워크를 세분화하여 각 클러스터들을 슈퍼피어라는 특정 피어가 일반 피어들에 대한 정보를 관리하는 형태이다[14,15]. 검색이 소수의 슈퍼피어들에게만 요청되어지기 때문에 네트워크 부하가 적고, 각 클러스터에 대한 정보를 슈퍼피어가 보유, 관리하기 때문에 검색의 효율성도 높다. 이러한 슈퍼피어 기반 P2P 네트워크에서의 트래픽 비용에 영향을 주는 요소 중 하나가 전체 네트워크에 존재하는 슈퍼피어의 개수이다. 슈퍼피어 기반 P2P네트워크에서 너무 많은 슈퍼피어가 존재한다면 각 클러스터내의 네트워크 트래픽은 적을지 모르나, 한 슈퍼피어가 보유하고 있는 정보의 양이 적어서 일반 노드의 요청에 답하지 못해 다른 슈퍼피어에게 요청할 수밖에 없기 때문에 슈퍼피어 간 메시지양은 더 증대될 것이다. 반대로 슈퍼피어 개수가 너무 적다면 슈퍼피어 간 메시지양이 줄어드는 대신 클러스터 내 트래픽 양이 많아질 것이다.

본 논문에서는 이러한 슈퍼피어 기반 P2P네트워크에서 슈퍼피어의 개수 변화에 따른 트래픽 비용을 연구하였다. 다양한 네트워크 크기에서 슈퍼피어 개수 변화에 따른 트래픽 비용의 변화를 클러스터 내의 비용과 클러스터 간의 비용으로 측정하고, 이 비용이 최소가 되는 슈퍼피어의 최적 개수를 제시할 것이다. 네트워크에서의 트래픽 비용 구성은 연구의 목적에 따라 가장 크게 영향을 미치는 요인들이 다를 수 있으나, 본 논문에서의 트래픽 비용은 발생되는 메시지의 개수 및 크기와 함께 이들의 라우팅 즉, 메시지의 전달이 클러스터 내로 국한 될 때와 다른 클러스터로 넘어가는 경우를 포함하여 구성되며 상세한 내역은 3장에서 다룰 것이다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로 또 다른 구조적 P2P 네트워크인 DHT기반 P2P네트워크의 기본 개념과 DHT기반 P2P 네트워크의 대표적인 알고리즘들을 소개하고 슈퍼피어기반 P2P네트워크의 구조와 동작원리를 알아본다. 3장에서는 트래픽 비용계산을 위한 클러스터 내 트래픽 비용, 슈퍼피어 간 트래픽 비용, 그리고 앞의 두 비용을 토대로 얻어진 전체 트래픽 비용을 수식으로 정리한 후 클러스터의 크기를 다양하게 변화시킨 시뮬레이션의 결과 값과 비교해 본다. 4장에서는 수식 모델로부터 얻을 수 있는 몇 가지 분석 결과를 언급하고, 5장에서 논문의 결론과 향후과제를 제시한다.

2. 관련연구

DHT(Distributed Hash Table) 기반 P2P 네트워크에서 분산해쉬 테이블은 일반 해쉬 테이블을 응용 프로그램 계층에서 논리적으로 생성되어 물리적인 변화에도 상관없이 동작하는 오버레이 네트워크에 적용한 것이다[9]. 각 피어들은 <key,value>쌍의 일부를 저장하고 있고, 전체 네트워크는 커다란 해쉬 테이블로 표현된다. key는 파일의 이름을 해쉬하여 얻은 것이고, value는 해쉬한 파일을 갖고 있는 피어에 대한 정보이다. (그림 2)에서는 각 피어의 아이디와 파일의 키를 얻는 과정을 보여주고 있다. 각 피어들은 아이디를 갖게 되는데, 그 아이디는 피어의 주소를 해쉬한 값이다. 그리고 각 피어들은 자신이 가지고 있는 파일의 이름을 해쉬하여 키를 구한다. 아이디와 키는 모두 숫자값으로 구해진다. 파일 이름을 해쉬하여 키를 얻으면 그 파일을 갖고 있는 피어는 네트워크에서 이 키를 담당할 피어를 선택하여 이 피어에게 키에 대한 정보를 전달한다.



(그림 2) 피어ID와 key를 얻는 과정

DHT 기반 P2P 네트워크의 파일 검색은 찾고자 하는 파일 이름을 해쉬하여 얻은 키를 이용한다. 우선 그 키를 담당하고 있어 키에 대한 정보를 가지고 있는 피어를 찾는다. 키를 저장하고 있는 피어는 키에 해당하는 파일을 저장하고 있는 피어에 대한 정보도 갖고 있다. 이 정보를 이용하여 키 값에 해당하는 파일을 갖고 있는 피어와 직접 연결하여 그 피어에게 요청 메시지를 유니캐스팅(unicasting)하여 원

하는 파일을 전달받게 되는 것이다.

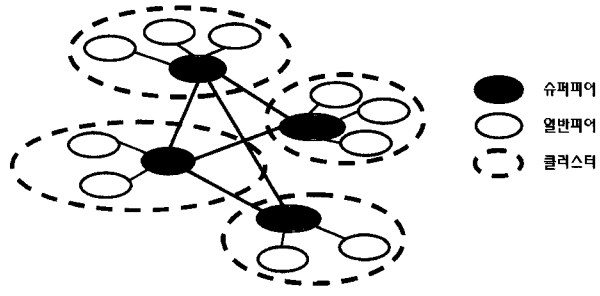
DHT를 기반으로 하는 구조적 P2P 알고리즘 중의 하나인 Pastry는 각 피어ID의 Prefix를 이용하여 라우팅하는 알고리즘이다[10]. Pastry의 피어들은 128비트 피어ID와 공유 파일의 키를 갖게 되고, 파일의 키는 그 키 값과 숫자적으로 가까운 피어ID의 노드에 저장되어진다. 각 Pastry의 피어들은 라우팅 테이블과 Leafset, 이웃노드세트를 보유하고 있고, 이 정보들을 이용하여 검색을 실행한다. 키 값을 검색할 때에는 우선 키가 leafset의 범위에 있는지 살펴본다. 키 값이 leafset 범위 안에 있으면 leafset에 저장되어 있는 피어 중 ID가 키와 가장 근접한 피어로 검색이 진행된다. 키가 leafset의 범위에 없다면 라우팅 테이블을 체크해서 가능한 한 긴prefix를 공유하는 ID로 검색을 진행시킨다.

Tapestry는 노드ID의 suffix를 이용하여 라우팅하는 DHT 기법이다[11]. 각 피어와 객체들은 160비트 식별자(identifier)들이 해쉬함수를 통해 할당되며 객체에는 GUID가 할당되어진다. 각 키는 키 값과 같거나 최소한 하나이상의 suffix가 동일한 root 피어에 저장된다. Tapestry의 neighbor map은 동일 suffix 수에 따라 L1, L2등의 레벨이 있고 레벨 수만큼 suffix를 공유하는 피어들을 저장하고 있다. Neighbor map을 사용하여 원하는 key 검색 시에 key와 가장 긴 suffix를 공유하는 노드로 경로 설정이 이루어진다.

CAN은 D차원의 가상공간을 가지는 네트워크 구조이다[12]. 키 값은 d차원에서 한 점에 대응되는 벡터로 구해지고, 각 노드는 ID 좌표 값에 따라 하나의 존(zone)에 대응되어 존의 범위에 해당하는 데이터를 저장한다. 만약 두 개의 노드가 속한 좌표 값 중 d-1 차원의 값이 중첩되며 한 개 차원이 이웃하면 이웃 노드라 한다. 그리고 각 피어들은 이러한 이웃 노드들의 존 정보와 IP주소를 가진 라우팅 테이블을 갖는다. 키 검색은 각 노드가 가지고 있는 이웃노드 정보 테이블에 있는 이웃 노드 중 키 벡터 값에 근접한 존으로 진행하여 원하는 키 값을 가지고 있는 노드에 도달함으로써 이루어진다.

Chord는 노드와 해쉬 값을 위해 원형의 시계방향으로 노드가 진행되고 m-bit식별자 공간을 사용하는 네트워크이다[13]. 해쉬 함수에 의해 얻어진 키 값은 같은 값을 가지는 노드나 키 값보다 더 큰 노드 중 첫번째 노드에 저장된다. 이 노드를 successor노드라고 한다. 각 노드는 라우팅 테이블로 finger table을 가지고 있는데 이 테이블에는 노드의 위치를 기준으로 +2i만큼씩 시계방향으로 거리를 지수적으로 증가시키고 그 범위에 있는 successor노드를 저장하고 있다. 검색 라우팅은 finger table의 정보를 따라 범위를 지수적으로 감소시키면서 목적지 노드를 찾게 된다.

DHT기반 P2P 네트워크의 특징은 N개의 노드가 존재한다고 했을 때, 검색 시 O(logN)의 홉 수를 보장한다는 것이다. 각 노드가 가지고 있는 네트워크에 대한 정보 테이블을 이용하여, 검색 메시지는 단계적으로 원하는 파일 키 값 정보를 가지고 있는 노드에 도달하게 되며 이 때의 홉 수가 O(logN)이다. 그러므로 검색 진행에 따른 네트워크 부하도 적고, 빠르고 정확한 검색이 가능하다[10~13]. 그러나 파일의 키 값으로 검색을 진행하는 DHT기반 P2P 네트워크 모델은 검색하고자 하는 파일의 키 값을 정확히 알지 못한다면 검색의 수행이 어렵다. 네트워크 상의 한 홉은 실제로는 여러 네트워크 망을 경유할 수도 있는데도 불구하고, DHT 기반 P2P 네트워크는 이러한 물리적인 상황을 고려하지 않는다[14]. 이러한 DHT기반 P2P 네트워크의 문제점을 개선하기 위한 방안이 슈퍼피어 기반 P2P 네트워크에서 진행되고 있으며, (그림 3)은 간략한 슈퍼피어 기반 P2P 네트워크의 구성도이다[14~18].



(그림 3) 슈퍼피어 기반 네트워크 구성도

- 슈퍼피어(Super-peer) : 서브 네트워크의 일반 피어들의 정보를 관리하는 피어이다. 슈퍼피어는 두 개의 테이블을 가지고 있다. 하나는 자신이 관리하고 있는 클러스터 내의 피어와 그 피어가 가지고 있는 공유 파일 정보에 대한 테이블이다. 또 하나는 네트워크 내의 슈퍼피어 목록 테이블이다.
- 일반피어(Client Peer) : 슈퍼피어의 관리를 받는 피어이다. 네트워크 가입(join)시 새로운 피어는 우선적으로 슈퍼피어에게 등록해야 하고 공유하기 원하는 파일에 대한 정보를 슈퍼피어에게 전달해주어야 한다. 일반피어는 자신이 가입되어 있는 슈퍼피어에 대한 정보를 가지고 있어야 한다.
- 클러스터(Cluster) : 하나의 슈퍼피어와 하나 이상의 일반피어로 이루어진 서브 네트워크를 말한다.

[14]에서는 리더들로 구성되는 슈퍼 네트워크와 물리적으로 가까운 노드들로 구성된 서브 네트워크로 이루어진 Grapes라는 P2P 네트워크 모델을 소개하고 있다. Grapes는 새로운 데이터를 네트워크에 삽입하는데 걸리는 시간, 필요한 데이터를 검색하는데 걸리는 시간, 피어가 원하는 데이터를 전송받는데 걸리는 시간 등이 DHT기반 P2P 네트워크보다 빠른 속도로 수행됨을 보이고 있다. Grapes의 경우 각 동작의 수행 시간을 측정하고 있기 때문에 네트워크 트래픽이 구체적으로 어떻게 변화하는지는 알기 힘들다.

[15]는 비구조적인 P2P 네트워크의 과도한 트래픽을 해소하기 위한 슈퍼피어 네트워크이다. 트래픽 비용과 검색에 대한 결과 개수, 검색을 요청한 피어가 최종 답변을 얻게 되기까지의 평균 홉 수 등의 결과를 보여주고 있다.

[16]은 O(1)이라는 고정 시간 내에 검색이 수행되는 것을 목적으로 한 네트워크 모델에 대한 연구이다. DHT기반 P2P 네트워크 중 Chord를 클러스터링해서, 각 클러스터를 관리하는 슈퍼피어들이 서로 링 형태로 연결되어 메시지를 주고받는 형태이다. 슈퍼피어가 클러스터를 관리하는데 필요한 메모리양, 검색 진행, 클러스터 유지를 위한 트래픽등을 수식으로 표현하였고, 시뮬레이션 결과 값과 수식으로 표현된 동작 결과 값들이 거의 일치함을 보이고 있다.

본 논문에서는 슈퍼피어 개수에 따라 변화하는 네트워크 트래픽 양을 측정하여 네트워크의 크기에 따른 최적의 슈퍼피어 개수를 예측하고자 한다. 일반적으로 트래픽 비용을 언급할 경우 메시지의 양뿐만 아니라 개별 피어간의 거리에 따른 전송지연, 스위칭 방식, 대역 폭 등과 같은 다양한 요인을 모두 포함시켜야 하지만 이 경우 예상되는 모든 환경을 수식을 통한 분석모델로 접근하기가 거의 불가능하다. 예를 들어 피어간의 거리를 따지는 경우의 수조차 모델링의 범위를 가늠하기 힘들 것이기 때문이다. 따라서 본 논문에서는 발생되는 전체 메시지의 양을 기반으로 트래픽 비용을 정의하고 네트워크의 크기에 따라 최소의 메시지 양을 보이는 클러스터 개수를 추정한다. 수학적 모델링을 위해 클러스터의 크기를 동일하게 가정하였으나 서로 다른 크기의 경우에 대한 시뮬레이션 결과를 보임으로써 메시지 양에 관한 한 트래픽 비용은 클러스터 크기의 다양함이 아닌 개수에 달려있음을 보일 것이다. 결과적으로 네트워크를 클러스터링할 경우 클러스터 각각의 크기에 대한 부담으로부터 벗어나 제시된 수식 모델에 의해 최적의 클러스터 개수를 쉽게 구할 수 있을 것이다.

본 논문은 앞에서 열거된 선행 연구들과 비용 연구의 방향이 다름으로 인해 직접적인 결과 비교는 쉽지 않은 아쉬움이 있으나 이상의 논문들은 슈퍼피어 기반 P2P 네트워크 환경에서 시도된 비용관련 연구라는 측면에서 살펴보았으며 기본적인 가정과 파라미터 중에서 본 논문과 같은 환경은 참고, 인용하였다.

3. 슈퍼피어 기반 P2P 네트워크의 트래픽 비용

슈퍼피어 기반 P2P 네트워크에서 발생되는 트래픽은 피어 가입, 정보 갱신, 정보 검색의 세 가지 동작에 의해 구성된다.

새로운 피어가 네트워크에 들어온다면, 이 피어는 하나의 슈퍼피어에게 등록이 되어야 한다. 우선 새로운 피어는 기존의 피어 중 근처에 있는 피어와 통신하고, 기존의 피어는 자신이 갖고 있는 슈퍼피어 정보를 새로운 피어에게 알려준다. 기존의 피어에게 받은 그 정보로 새로운 피어는 슈퍼피어에게 가입요청 메시지를 보내고, 슈퍼피어는 새로운 피어에게 ID를 부여하여 피어 목록 테이블에 저장한 다음 새로운 노드에게 가입 완료 메시지를 보낸다. 이 때 새로운 노드가 등록을 원하는 파일이 있다면 슈퍼피어는 그 파일에 대한 정보도 저장한다[16-18].

일반 피어 및 슈퍼피어는 주기적으로 서로의 상태를 체크한다. 이때의 상태 체크 정보는 체크하고자 하는 피어가 네트워크 상에 있는지 여부를 확인하는 것이다[16].

일반 피어는 원하는 파일 검색 요청을 자신이 등록되어 있는 슈퍼피어에게 보낸다. 슈퍼피어는 요청된 파일에 대한 정보를 자신이 가지고 있는지 검색한다. 만약 보유하였다면 그 파일의 <FileName, 피어ID, 피어의 주소>를 보내준다. 보유하지 않았다면 다른 슈퍼피어에게 파일의 정보를 요청하고, 요청을 받은 슈퍼피어들은 자신의 테이블을 검색하여 요청한 파일이 있는 경우 그 결과를 요청한 슈퍼피어에게 전달한다[16~18].

트래픽 비용은 두 가지 영역으로 분리하여 계산할 수 있다. 첫 번째는 클러스터 내에서 피어가입, 정보갱신, 정보검색의 동작이 일어날 경우 발생하는 일반피어와 슈퍼피어 간에 주고받는 메시지 양을 측정한 비용이다. 그리고 두 번째는 정보검색 시 자신이 속한 클러스터의 슈퍼피어가 보유하지 않은 정보요청인 경우 다른 슈퍼피어와 주고받게 되는 메시지 양과 슈퍼피어간의 정보갱신을 측정한 비용이다.

$$Ctotal = \sum Ccluster + \sum Csuper$$

3.1 최적 클러스터 수를 위한 수식 모델

본 논문에서 네트워크의 트래픽 비용을 유도하기 위해 사용될 기본적인 파라미터들의 정의는 관련연구[14]를 참조하여 <표 1>과 같이 정리하였으며, 제한적인 측면은 있으나 측정치의 정확도를 위해 다음과 같이 가정하였다.

- 클러스터 내 일반 피어가 n개이면 슈퍼피어의 파일 목록 테이블도 n개이다.
- 클러스터 내의 피어 개수는 균등하게 분포하도록 한다.
- 슈퍼피어의 관리를 받지 않고 네트워크 내에서 독립적으로 동작하는 일반노드는 없다.
- 한 클러스터에는 하나의 슈퍼피어가 존재해야 한다. 만약 기존의 슈퍼피어가 네트워크 상에서 사라진다면 해당 클러스터의 일반 노드 중에서 슈퍼피어를 선출하도록 한다.
- 다른 슈퍼피어에 대한 정보는 주소만 가지고 있다.
- 모든 네트워크의 동작이 발생하는 시간적인 비율은 일정하다. 하지만, 네트워크 크기를 기반으로 하는 실험

〈표 1〉 네트워크 파라미터 정의

네트워크 크기	네트워크 내의 전체 피어 개수
클러스터 크기	슈퍼피어를 포함한 한 클러스터 내 노드 개수. 각 클러스터내의 피어 개수를 균등하게 한다고 했을 때 평균 클러스터 크기는 (네트워크 크기/슈퍼피어 개수)로 정의 됨.
검색발생 비율	개별 피어에 의해 단위 시간당 발생하는 검색 요청의 평균 값
정보갱신 비율	클러스터 당 단위 시간에 발생하는 정보갱신 요청의 평균 값
피어가입 비율	클러스터 당 단위 시간 발생하는 새로운 노드의 네트워크 가입 요청의 평균 값
피어탈퇴 비율	피어가입 비율과 동일

이기 때문에 네트워크 크기를 일정하게 하기 위해 노드의 피어가입 비율과 피어탈퇴 비율은 같다고 두며 이를 위해 전체 네트워크는 균형 상태(equilibrium state)라고 가정한다.

3.1.1 클러스터 내 트래픽 비용

계산에 필요한 식은 [16]과 [17]에서 동일한 환경인 부분을 선택적으로 참조하였고, 트래픽 비용 계산에 필요한 부분을 보완하여 추가한 변수 및 개체는 다음과 같다.

- N : 전체 네트워크의 피어 수
- S : 슈퍼피어 개수
- *Ccluster* : 클러스터 내의 트래픽 비용
- *Csuper* : 슈퍼피어 간의 트래픽 비용
- *Cq, Cj, Cu* : 정보검색, 노드가입, 정보 갱신의 비용
- *CQsend, CJsend, CUsend* : 정보검색, 노드가입, 정보갱신 시 메시지를 보낼 때 발생하는 비용
- *CQrev, CJrev, CUrev* : 정보검색, 노드가입, 정보갱신 시 결과를 전달받을 때 발생하는 비용
- *Qrate, Jrate, Urate* : 정보검색, 노드가입, 정보갱신의 단위시간 당 발생 비율(회수/초)
- *Qpacket, Jpacket, Upacket* : 정보검색, 노드가입, 정보갱신 시 메시지를 보낼 때의 패킷 길이
- *Qrec, Jrec, Urec* : 정보검색, 노드가입, 정보갱신 시 결과 값을 전달받을 때의 패킷 길이

클러스터 내 트래픽 비용을 계산하기 위한 필요한 수행동작은 위에서 정리한 대로 새로운 피어 가입과 정보 검색, 정보 갱신 세 가지이다. 따라서 클러스터 내 트래픽 비용은 이 세 가지 동작의 수행 시 드는 비용을 합산한 값이 된다.

$$Ccluster = Cq + Cj + Cu \tag{1}$$

(1)에서 *Cq*는 아래와 같은 식으로 표현할 수 있다.

$$Cq = CQsend + CQrev = Qrate \cdot Qpacket + Qrate \cdot Qrec \tag{2}$$

피어가 요청한 파일을 슈퍼피어가 보유하고 있는 경우라면 슈퍼피어가 즉각 응답하겠지만, 그렇지 않은 경우라면 슈퍼피어는 다른 슈퍼피어에게 검색을 요청한다. 다른 슈퍼피어에게 검색을 요청하는 것은 클러스터 외부의 비용이지만, 결과 메시지를 받는 슈퍼피어는 결국 자신의 클러스터 내의 일반 노드에게 그 결과 메시지를 전송하기 때문에 일반피어에게 정보검색이 들어온 비율만큼 결과 메시지도 보내게 된다. 정보검색이 피어로부터 단위 시간당 q의 단위로 발생한다면 클러스터 당 평균 검색비율은 $Qrate = q \cdot (N/S)$ 로 표현되므로 *Cq*는 다음과 같은 식 (3)과 같이 되며, 정보 갱신의 경우 슈퍼피어가 클러스터내의 피어들에게 갱신을 위한 메시지를 동시에 보낸 후 각 피어들로부터 수신을 받게 되므로 식 (5)와 같다.

$$Cq = q \cdot (N/S) \cdot Qpacket + q \cdot (N/S) \cdot Qrec \tag{3}$$

*Cj, Cu*는 정리하면 다음과 같다.

$$Cj = CJsend + CJrev = Jrate \cdot Jpacket + Jrate \cdot Jrec \tag{4}$$

$$Cu = (N/S) \cdot (CUsend + CUrev) = (N/S) \cdot (Urate \cdot Upacket + Urate \cdot Urec) \tag{5}$$

*Jpacket*와 *Jrec* , *Upacket*와 *Urec*은 각각 크기가 같기 때문에 (4)번과 (5)번 식은 각각 (6)번과 (7)번 식으로 간단하게 표현할 수 있다.

$$Cj = 2 \cdot Jrate \cdot Jpacket \tag{6}$$

$$Cu = 2 \cdot (N/S) \cdot Urate \cdot Upacket \tag{7}$$

따라서 식 (3), (6), (7)을 사용하여 클러스터 내의 트래픽 비용에 대한 수식 (1)을 정리하면 다음과 같다.

$$Ccluster = q \cdot (N/S) \cdot Qpacket + q \cdot (N/S) \cdot Qrec + 2 \cdot Jrate \cdot Jpacket + 2 \cdot (N/S) \cdot Urate \cdot Upacket \tag{8}$$

3.1.2 클러스터 간 트래픽 비용

클러스터 간에 - 클러스터의 개수는 곧 슈퍼피어의 개수이므로 이 후 두 용어는 같은 의미로 사용한다. - 발생하는 비용도 정보검색과 정보갱신 시 발생하는 메시지의 양을 의미하므로 이 비용을 위해 슈퍼피어 간에 주고받는 메시지 양을 측정 한다. 슈퍼피어 간 메시지 발생은 슈퍼피어가 요청된 파일에 대한 정보를 가지고 있지 않은 경우에 발생하기 때문에 다음과 같은 확률 값을 고려하여야 한다.

- *Pr[Qsuc]* : 일반 노드가 정보를 검색할 때 자신이 속한 클러스터의 슈퍼피어에서 성공할 확률.
- *Pr[Qfail]* : 일반 노드가 정보를 검색할 때 자신이 속한 클러스터의 슈퍼피어에서 실패할 확률.

네트워크상의 모든 피어들이 하나 이상의 파일을 공유하는 상황이라면 피어의 개수가 많은 클러스터의 슈퍼피어의 검색 성공확률이 피어의 개수에 비례하여 커지게 됨은 자명하나, 여기서는 각 클러스터가 같은 개수의 피어를 가지는 것으로 가정하였으므로 *Pr[Qsuc]*와 *Pr[Qfail]*은 다음과 같다. 참고로 각 슈퍼피어가 가지는 파일은 위의 가정에서 언급한대로 자신이 관리하는 피어들에 대한 목록 테이블로서 피어의 개수에 비례하는 크기이며 단위 크기를 고려한 실제 크기를 위한 상수 배는 분자와 분모에 공통으로 있어야하나 확률 값에 영향을 주지 못하므로 생략하였다.

$$Pr[Qsuc] = \text{하나의 슈퍼피어가 소유한 파일크기} / \text{슈퍼피어들이 소유한 파일 크기 합} = (N/S) / N = 1/S$$
$$Pr[Qfail] = 1 - Pr[Qsuc]$$

슈퍼피어 검색실패로 인한 슈퍼피어 간의 트래픽 비용은, 검색실패로 인해 나머지 슈퍼피어에게 검색요청을 보내는 비용과 검색요청의 응답 비용 그리고 다른 슈퍼피어의 상태를 체크하는 정보갱신의 합이라고 할 수 있다. 식 (9)는 슈퍼피어 간의 트래픽 비용을 식으로 나타낸 것이다.

$$\begin{aligned} Csuper &= Qrate \cdot Pr[Qfail] \cdot [Qpacket \cdot (S-1) + Qrec \cdot (S-1)] + 2 \cdot Urate \cdot Upacket \cdot (S-1) \\ &= q \cdot (N/S) \cdot (S-1)/S \cdot [Qpacket \cdot (S-1) + Qrec \cdot (S-1)] + 2 \cdot Urate \cdot Upacket \cdot (S-1) \end{aligned} \quad (9)$$

3.1.3 슈퍼피어 개수에 따른 트래픽 비용

슈퍼피어 개수에 따른 전체 트래픽 비용은 앞 절에서 얻은 클러스터 내 트래픽 비용과 클러스터 간 트래픽 비용에 대한 식에서 구할 수 있다. 전체 트래픽 비용에서는 슈퍼피어가 관리하는 일반 노드 테이블을 위한 메모리 비용과 검색을 위한 테이블 탐색 비용과 같은 트래픽과 무관한 부분은 제외하였다.

전체 트래픽 비용을 구하기 위해서는 위의 식 $Ctotal = \Sigma Ccluster + \Sigma Csuper$를 그대로 사용하여야 하지만 본 논문에서는 모든 클러스터내의 피어 개수가 모두 같은 상황으로 가정하고 있기 때문에 클러스터 내 트래픽 비용의 전체 합은 클러스터 개수*클러스터 내 트래픽 비용과 같고, 슈퍼피어 간 트래픽 비용의 전체 합 또한 클러스터 개수* 슈퍼피어 간 트래픽 비용과 같게 된다. 즉, 두 비용 모두 클러스터 개수라는 같은 수를 곱하기 때문에 전체 트래픽 비용은 하나의 클러스터 내 트래픽 비용과 하나의 슈퍼피어 간 트래픽 비용의 합을 정수배한 것이 되며 이것은 최적 슈퍼피어 개수를 구하는 데는 영향을 미치지 않음을 알 수 있다. 따라서 전체 트래픽 비용을 하나의 클러스터 내 트래픽 비용과 하나의 슈퍼피어 간 트래픽 비용의 합으로 전개한 후 전체 네트워크에서의 최적 슈퍼피어 개수를 구할 수 있다.

$$\begin{aligned} Ctotal &= Ccluster + Csuper \\ &= [q \cdot (N/S) \cdot Qpacket + q \cdot (N/S) \cdot Qrec + 2 \cdot Jrate \cdot Jpacket + 2 \cdot (N/S) Urate \cdot Upacket] + \\ &\quad [q \cdot (N/S) \cdot (S-1)/S \cdot \{Qpacket \cdot (S-1) + Qrec \cdot (S-1)\} + 2 \cdot Urate \cdot Upacket \cdot (S-1)] \end{aligned} \quad (10)$$

식 (10)을 만족하는 S의 최소 값이 최적 슈퍼피어 개수가 되며 이것은 식(11)을 만족하는 S값과 같다.

$$\frac{dCtotal}{dS} = 2 \cdot U_{rate} \cdot U_{packet} \cdot S^3 + (q \cdot N \cdot Q_{packet} \cdot Q_{rec} - 2 \cdot U_{rate} \cdot U_{packet} \cdot N) \cdot S - 2 \cdot q \cdot N \cdot Q_{packet} \cdot Q_{rec} = 0 \quad (11)$$

S의 3차식인 식 (11)은 근을 바로 구할 수 없기 때문에 수치해석의 도구를 사용해 근사 값을 찾아야하며 네트워크의 크기에 따른 최적 S값들은 다음 절의 그래프들과 <표 3>으로 정리하였다.

3.1.4 수식 모델에 의한 트래픽 비용 분석

3.1절의 수식을 토대로 실제 파라미터 값을 도입하여 트래픽 비용을 계산하고, 그 결과를 분석하여 다양한 네트워크 크기에서의 슈퍼피어 개수 변화에 따른 트래픽 비용 변화를 구하였다. 슈퍼피어와 일반 피어가 동작하는 각 수행동작의 비용 - 발생되는 메시지의 크기 - 는 <표 2>와 같으며 사용된 값들은 앞서 관련연구에서 언급된 논문들로부터 유사 또는 동일한 환경에 사용된 값들을 준용하였다.

〈표 2〉 각 동작의 비용

Action	Bandwidth Cost(Bytes)
Send query Process Query Send Response	82+query length 0 80+28*address+76*number of results
Send Join Receive Join Process Join	80+72*number of files 80+72*number of files 0
Send Update Receive Update Process Update	152 152 0

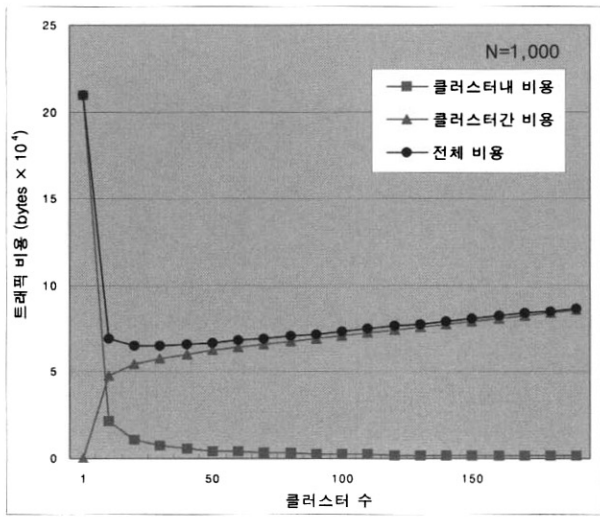
Send query는 정보검색 시 쿼리를 보낼 때의 동작을 말하고 그 패킷 길이는 기본 패킷 길이 82에 요청 파일의 쿼리 길이가 된다. Send Response는 정보검색에 대한 응답으로써, 기본 패킷 길이 80에 찾고자하는 파일을 가지고 있는 피어의 주소 값에 28을 곱한 값과 찾아낸 파일의 개수에 76을 더한 값이 전체 패킷 길이가 된다.

Send Join은 새로운 노드가 처음 네트워크에 도착했을 때 클러스터를 관리하고 있는 슈퍼피어에게 보내는 패킷을 보내는 동작으로 기본 패킷 길이 80에 공개하길 원하는 파일의 개수에 72를 곱한 값이 패킷 길이이다. Receive Join은 슈퍼피어가 새로운 노드가 등록이 됐음을 알리는 동작으로서 패킷 길이는 Send Join의 길이와 동일하다.

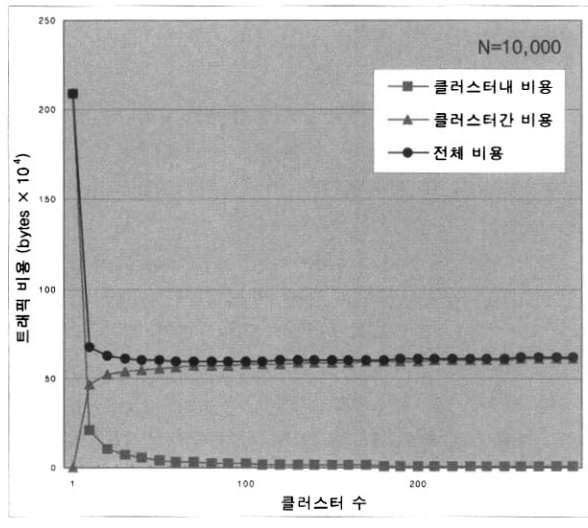
Send Update는 슈퍼피어가 클러스터내의 일반피어와 다른 슈퍼피어와의 연결 상태를 체크하는 동작으로 152라는 일정한 패킷 길이를 가지고 있다.

<표 2>의 각 동작의 비용에서 query length는 [14]에서 나온 평균 길이를 참고하여 12로 하였고, address 길이는 32로 하였다. number of results와 number of files는 하나의 파일에 대한 복사본이 네트워크 상에 없다는 가정 하에 1로 하여 실험을 진행하였다.

분석을 위해 전체 피어의 개수가 각각 10^3, 10^4, 10^5인 네트워크를 대상으로 슈퍼피어 개수를 2승씩 증가시킨 상황의 트래픽 비용을 계산하였다. [16]에서 사용한 값을 준용하여



(그림 4) N=1,000일 때의 최적 클러스터 수 추정 그래프



(그림 5) N=10,000일 때의 최적 클러스터 수 추정 그래프

q값은 0.005, $Jrate$값은 1.0, $Urate$값은 0.05로 각각 설정하여 계산된 그래프들과 함께 q와 $Urate$값을 변화시켰을 각각의 경우에 대해 얻어지는 최적의 슈퍼피어 개수는 <표 3>에서 정리하였다. 식 (11)에서 알 수 있듯이 $Jrate$값은 영향을 미치지 못하므로 1.0으로 고정하였다.

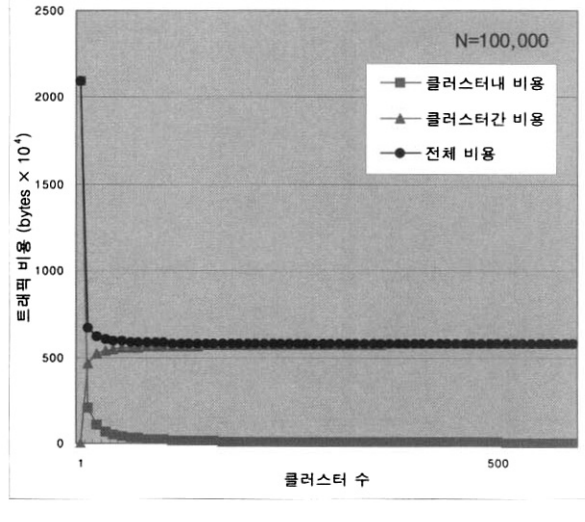
(그림 4)는 피어의 개수가 1000인 크기의 네트워크에서 $Ccluster$ 비용과 $Csuper$ 비용의 변화에 따른 전체 트래픽 비용을 나타내며, 이를 최소로 하는 그래프 상의 위치가 최적의 클러스터 수 즉, 최적의 슈퍼피어 개수임을 알 수 있다. 슈퍼피어 개수가 늘어남에 따라 클러스터 내의 비용이 줄어듦을 볼 수 있으며 이것은 슈퍼피어 개수가 증가하면 그만큼 관리하게 되는 일반 피어 개수가 줄어들기 때문에 클러스터 내 트래픽 비용 또한 줄어들게 된다는 것을 말하며 상대적으로 클러스터 간의 비용은 증가함을 의미한다.

(그림 5)와 (그림 6)은 N의 크기를 각각 10000, 100000으로 했을 때의 그래프이며 비용은 특정 위치에서 최소 값을 보인 후 N이 커질수록 증가하게 되는데, 그림 상으로는 최적 위치에 초점을 맞추기 위해 가로 축의 전체 구간을 표시하지 않음으로 인해 증가의 정도가 확연히 드러나지 않는다. 참고로 (그림 5)와 (그림 6)에서 최소 비용을 보이는 클러스터의 개수는 <표 3>에 있다.

(그림 7)은 네트워크의 크기가 선형 증가할 경우 최적 슈퍼피어 개수가 증가하는 정도를 보여주는 그래프이며 이를

〈표 3〉 Rate 변화에 따른 최적 슈퍼피어 개수

Network Size	Query Rate	Update Rate	Number of Optimal Cluster
1,000	0.005	0.1	29
		0.5	31
		1	31
	0.01	0.1	25
		0.5	30
		1	31
	0.05	0.1	4
		0.5	26
		1	29
10,000	0.005	0.1	90
		0.5	98
		1	99
	0.01	0.1	79
		0.5	96
		1	98
	0.05	0.1	4
		0.5	79
		1	90
100,000	0.005	0.1	285
		0.5	310
		1	313
	0.01	0.1	250
		0.5	304
		1	310
	0.05	0.1	4
		0.5	250
		1	285



(그림 6) N=100,000일 때의 최적 클러스터 수 추정 그래프

토대로 네트워크의 크기에 따른 최적 슈퍼피어 개수는 쉽게 추정이 가능하다.

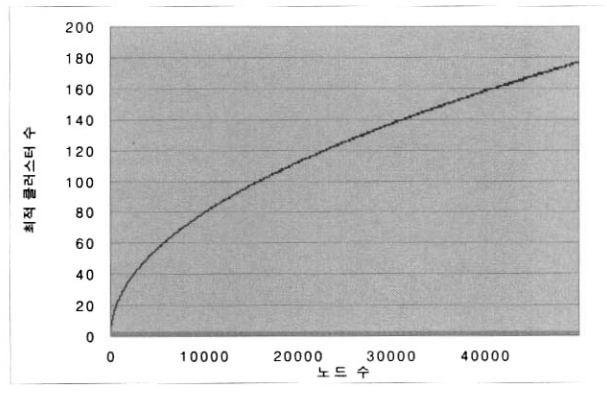
3.2 크기가 다른 클러스터링의 시뮬레이션

3.1절에서는 수식 모델을 위해 각 슈퍼피어가 관리하는 일반 피어의 개수가 동일하다고 가정하였다. 현실적으로 매우 다양한 크기의 클러스터가 존재하는 상황에서 이 가정이 유효하기 위해서는 클러스터의 크기를 서로 다르게 한 시뮬레이션 모델을 통해 전체 트래픽 비용이 클러스터 당 피어 개수의 상이함에 좌우되기보다는 클러스터의 개수에 달려 있음을 보여야 할 것이다.

하나의 피어는 하나의 슈퍼피어와 통신하므로 3.1절의 수식들과 (그림 7)로부터 알 수 있듯이 메시지의 양에 근거한 트래픽 비용은 기본적으로 사용된 비율 값에 영향 받으며 네트워크의 전체 피어 개수 N에 따라 증가하게 될 것이다. 다시 말해 최소의 비용은 클러스터를 어떤 크기로 나누느냐가 아니라 몇 개로 나누느냐에 달려있음 예측할 수 있으며 이러한 사실은 시뮬레이션의 결과로 뒷받침 될 것이다.

전체 피어의 개수는 수식 모델과의 비교를 위해 동일하게 10^3, 10^4, 10^5인 네트워크를 대상으로 하였으며, 클러스터의 개수 당 난수 생성(random number generation)을 사용하여 5개의 서로 다른 모형을 적용하였다. 예를 들어, N이 1000이고 클러스터 개수가 10이라면 수식모델에서는 각 클러스터의 크기가 100으로 동일한 한가지뿐이지만 시뮬레이션에서는 10개의 클러스터 각각의 크기를 난수를 통해 다르게 만들되 그 가지 수를 5개로 하였다. 이럴 경우 클러스터 10개의 크기가 서로 비슷한 경우부터 매우 다른 크기가 섞여 있는 경우까지를 어느 정도 표현할 수 있을 것으로 판단되며 참고로 실제 실험에서는 모형의 수를 더 많이 주어 해보았으나 결과는 별 차이가 없었다.

클러스터로 나눌 때 각 피어들은 1부터 네트워크의 크기

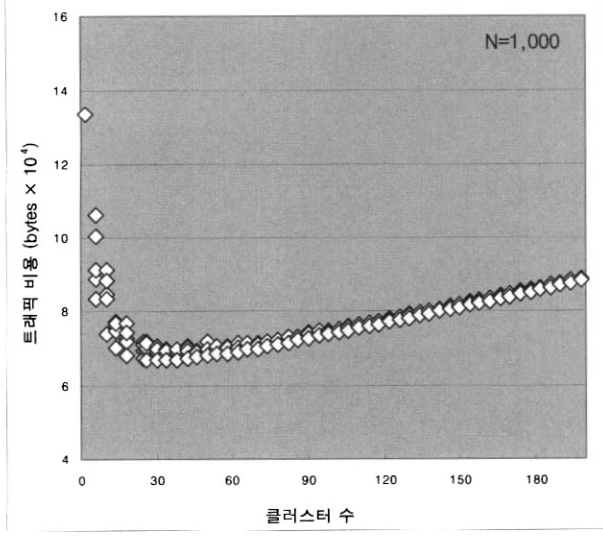


(그림 7) 네트워크 크기에 따른 최적 슈퍼피어 수

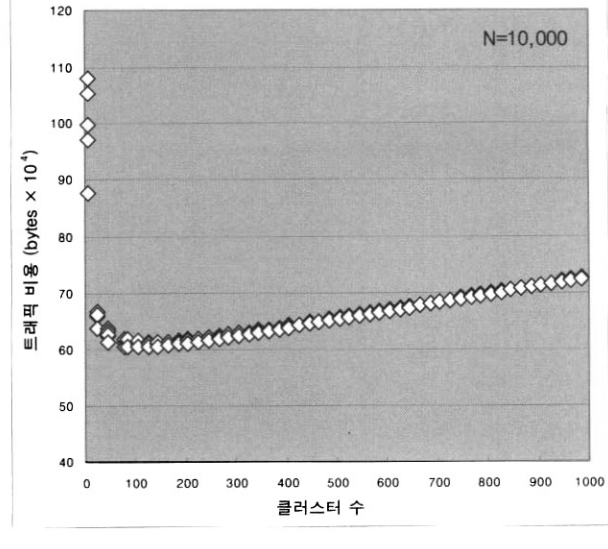
값 사이의 고유 값 을 가지도록 함으로써 자신이 속한 클러스터가 구별되도록 하였으며 클러스터들 역시 1부터 나눈 개수 값 사이의 고유 값으로 구분하여 몇 번 피어가 몇 번 클러스터에 속하는지가 분명히 드러나도록 하였다.

검색 쿼리의 발생은 비율에 맞춰 두 개의 피어에 대응되는 난수 두 개를 생성시켜 가면서 두 값이 동일한 클러스터에 속한 경우 클러스터 내의 비용으로, 다른 값일 경우 -서로 다른 클러스터에 속하는 값일 경우 - 클러스터간의 비용을 발생시킨 것으로 처리하여 수식에서의 *Pr[Qfail]*을 자연스럽게 반영할 수 있게 하였다.

(그림 8)부터 (그림 10)은 <표 2>의 값들과 발생 비율들은 수식 때와 동일하게- *q*값은 0.005, *Jrate*값은 1.0, *Urate* 값은 0.05로 각각 설정-적용하되 횟수를 누적시킨 후 경과 시간을 나눔으로써 단위 시간당 비용을 산출하여 그려진 그래프이며 세로축(클러스터 수)의 각 포인트마다 5개의 가로축(트래픽 비용)과 평행한 점들이 보이는데 이것은 위에서 말한 5개의 모형들 각각을 나타내는 점이며, 클러스터 수가 커질수록 겹쳐져 있음을 알 수 있다.

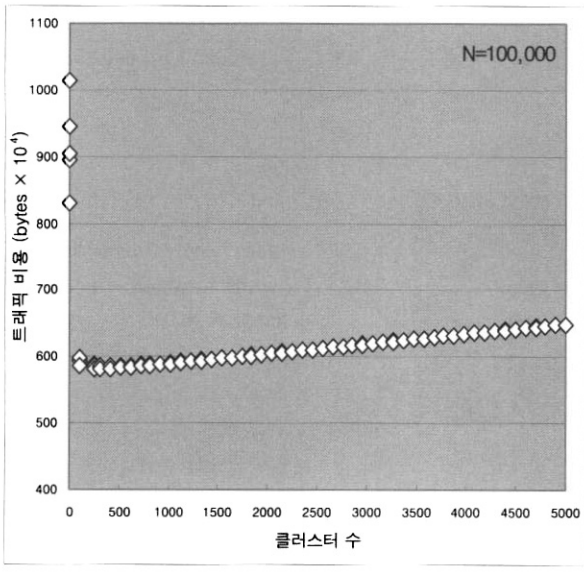


(그림 8) N=1,000일 때의 시뮬레이션 결과



(그림 9) N=10,000일 때의 시뮬레이션 결과

수식 모델의 결과 <표 3>에서 N이 1000일 경우 최적의 슈퍼피어 개수가 31인 것과 비교해 보면 (그림 8)에서 보이는 값 역시 매우 근사함을 알 수 있으며 이 점은 (그림 9)와 (그림 10)에서도 마찬가지이다. 참고로 (그림 10)과 같이 가로축의 포인트 값이 커질 경우 최적 클러스터 수를 나타내는 위치가 쉽게 알아보기 힘들 수 있으나 실험의 결과 값은 <표 3>에서 주어진 310과 근사하였다.



(그림 10) N=100,000일 때의 시뮬레이션 결과

4. 결과 분석

네트워크에서 트래픽 비용에 영향을 미치는 요인은 다양하다. 스위칭이나 대역 폭과 같은 요인은 하드웨어에 의존적이므로 논외로 하고, 각 피어 간의 거리에 의한 지연은 무선의 경우 무시해도 좋을 정도의 차이를 보인다. 더구나 이 모든 요인을 수식 모델에 반영하기는 본문에서 밝힌 대로 불가능하기 때문에 여기서는 이 중 가장 영향을 미치는 요인인 메시지의 양에 한정하여 트래픽 비용을 추정하는 수식 모델을 구축하였다. 클러스터 각각의 크기가 얼마든지 다를 수 있는 현실을 시뮬레이션에 반영하여 같은 크기를 가정한 수식 모델의 결과와 근사함을 보임으로서 주어진 크기의 네트워크에서 최소의 트래픽 비용은 클러스터 당 피어 개수의 상이함에는 상관없이 클러스터의 개수에 달려 있음을 알 수 있다. 결과적으로 네트워크의 크기가 주어진 상황에서 본 논문에서 주어진 수식 모델에 값을 대입함으로서 쉽고 빠르게 최적의 클러스터 개수를 추정할 수 있을 것이다.

<표 3>의 값들을 토대로 미루어 보면 갱신 비율이 커질수록, 동일한 갱신 비율의 경우 검색 요청의 비율이 작아질수록 최적의 클러스터 개수는 증가함을 알 수 있으며, 네트워크의 크기 변화에 따라 그 값은 (그림 7)과 같은 증가 추세를 보임으로써 비교적 작은 크기의 네트워크에서 클러스터 개수에 따른 트래픽 비용이 민감함을 알 수 있다.

5. 결 론

현재 P2P 네트워크 모델에 대한 연구는 네트워크 망에 구조적인 특성을 부여하여 빠르고 정확한 검색과 네트워크 부하를 방지하기 위한 연구로 집중되고 있다. 구조적 P2P 네트워크 모델 중 하나인 슈퍼피어 기반 P2P 네트워크는 네트워크를 서브 네트워크로 나누어서 각 클러스터를 슈퍼피어라는 피어가 그 클러스터내의 일반피어들을 관리하고, 다른 슈퍼피어들과 소통하여 전체 네트워크가 연결되어지는 네트워크 모델이다. 이러한 모델에서는 슈퍼피어 개수가 전체 트래픽 비용을 결정하게 될 것이다.

본 논문은 슈퍼피어 기반 네트워크에서 서로 다른 크기의 네트워크 환경 하에 슈퍼피어 개수를 변화시켜 그에 따른 트래픽 비용의 증감과 최소비용을 보이는 슈퍼피어 개수를 추정하였다. 실험 결과 네트워크 크기가 증가함에 따라 트래픽 비용이 최소가 되는 구간도 변화함을 볼 수 있었으며, 수식을 통해 트래픽 비용에 정보 검색의 발생 비율 값 등도 영향을 미치는 요인임을 알 수 있었다. 시뮬레이션의 결과를 통해 주어진 크기의 네트워크에서 최소의 트래픽 비용은 클러스터 당 피어 개수의 상이함에는 상관없이 클러스터의 개수에 달려 있음을 알 수 있었다. 결과적으로 네트워크의 크기에 따라 변화하는 정보의 검색과 노드 가입, 갱신에 소요되는 값들을 반영하는 슈퍼피어의 최적 개수를 수식 모델을 통해 추정할 수 있을 것이다.

향후 연구과제로는 비용에 영향을 미치는 요인들을 가능한 추가하여 수용할 수 있는 수식 모델의 확장이 될 것이며 이 경우의 시뮬레이션 역시 병행되어야 할 것이다. 가능함을 전제로 시간적 비용의 측면에서 피어 간의 거리를 기반으로 지연시간을 모델링 할 수 있는 수식과 시뮬레이션을 통해 트래픽 비용을 예측하는 연구도 유용해 보인다.

참 고 문 헌

[1] R.Schollmeier. "A definition of peer-to-peer networking for the classification of peer-to per architectures and applications," Proc.IEEE Conference P2P 2001, Linkoping Sweden. August, 2001
[2] 김영진, 엄영익, "P2P 컴퓨팅 환경 기반의 디스커버리 기법", 정보과학회지, Vol.22, pp6-7, March, 2004.
[3] D.S. Milojicic, "Peer-to-Peer Computing," HP Technical Report, HP Labora-to-ries, March, 2002.
[4] Napster. http://www.napster.com
[5] D. Tsoumakos and N. Roussopoulos, "Adaptive Probabilistic Search for Peer-to-Peer Networks," Proc. of the 3rd IEEE International Conference on P2P Computing, September, 2003.
[6] Gnutella, www.gnutella.com
[7] I.Clarke, O. Sandberg, B.Wiley, and T, Hong. "Freenet: A Distributed Anonymous Information Storage and Retrieval

System" Lecture Notes in Computer Science, 2009:46-66, 2001.
[8] S.Androutsellis-Theotokis, "A survey of peer-to-peer file sharing technologies," Technical Report WHP-2002-03, Athens Univ.of Economics and Business, 2002.
[9] U. Wieder, M.Dahlin, "A Simple Fault tolerant distributed Hash Table", IPTPS 2003, Berkeley CA, Feburary, 2003.
[10] A. Rowstron and P. Druschel, "Pastry: Scalable, Distributed object location and routing for large-scale peer-to-peer systems," Proc. IFIP/ACM International Conference on Distributed Systems Platforms, November, 2001.
[11] B.Zhao, J.Kubiatowicz, and A.Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area loation 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," Proc. of ACM SIGCOMM, 2001.
[13] I.Stoca, R.Morris, D. Karger, F.Kaashoer and H. Balakrishnan, "Chord: A scalable peer-to-per lookup service for Internet application," in Proc. ACM SIGCOMM 2001.
[14] Kwangwook Shin, Seunghak Lee, Geunhwi Lim, H.Yoon, Joong Soo Ma, "Grapes : Topology-based Hierachical Virtual Network for Peer-to-peer Lookup Services," In Proceeding of the International Conference Parallel Processing Workshops, 2002.
[15] B.Yang, H.Garcia-Molina, "Designing a Super-Peer Network," Proc. of IEEE International Conference on Distributed Computing Systems (ICDCS), 2002.
[16] A.T.Mizrak, Y.Cheng, V.Kumar, and S.Savage "Structured Superpeers: Leveraging Heterogeneity to Provide Constant-Time Lookup," Computer Science and Engineering Division, Unv. of California, San Diego, 2003.
[17] S.Jain, R.Mahajan, D.Wetherall, and G.Borriello, "Scalable Self-Organizing Overlays," Computer Science and Engineering Division, Unv. of Washington, 2001.

김 성 희

e-mail : sunghee46.kim@samsung.com
2003년 2월 숙명여자대학교 컴퓨터과학과 학사
2006년2월 숙명여자대학교 일반대학원 컴퓨터과학과 석사
2006년2월 삼성전자 정보통신총괄 입사
현재 삼성전자 정보통신총괄 통신연구소 차세대단말팀 선행단말Lab. 근무

김 주 균

e-mail : jgkim@sookmyung.ac.kr
1985년 서울대학교 계산통계학과
1985년~1986년 DEC Korea 근무
1988년 서울대학교 계산통계학과 계산학 석사
1988년 서울대학교 계산통계학과 계산학 박사
1992년~현재 숙명여대 정보과학부 컴퓨터과학전공 교수
관심분야 : 운영체제, 성능평가, Caching Strategies, Flash Memory Technologies.

이 상 규

e-mail : sanglee@sookmyung.ac.kr
1989년 Univ. of Southern California 컴퓨터과학과(학사)
1991년 George Washington University 컴퓨터과학과(석사)
1995년 George Washington University 컴퓨터과학과(공학박사)
1995년~1996년 AC Technologies Inc. U.S.A. Software Engineer
1995년~1996년 Geroge Washington University 박사후 과정
1997년~현재 숙명여자대학교 정보과학부 컴퓨터과학전공 교수
관심분야 : Wireless Mesh Networks, Pervasive Computing, Ad-hoc Networks 무선통신, Mobile Computing, Internet Technologies

이 준 수

e-mail : jslee@sookmyung.ac.kr
1990년 서울대학교 계산통계학과(학사)
1994년 University of Pennsylvania 대학원 전산학(공학석사)
1994년~1999년 삼성전자(선임연구원)
2004년 University of Southern California 대학원 전산학(공학박사)
2005년~현재 숙명여자대학교 정보과학부 컴퓨터과학 전공 조교수
관심분야 : 프로토콜, 센서 네트워크, 임베디드 시스템, 운영체제