공유된 멀티캐스트 트리에서 센터 위치 결정을 위한 새 알고리즘

준회원 강 신 규*, 정회원 심 영 철**

New Center Location Algorithms for Shared Multicast Trees

Shin-kyu Kang* *Associate Member*, Young-chul Shim** *Regular Members*

요 약

PIM, CBT, BGMP 등과 같이 공유트리를 사용하는 멀티캐스트 라우팅 알고리즘에서 공유 트리 센터의 위치는 패킷 전달 지연 시간에 많은 영향을 미치게 된다. 본 논문에서는 센터 위치 결정을 위한 알고리즘 세 가지를 제안하고 시뮬레이션을 통하여 성능을 분석하였다. 이 세 알고리즘들은 멀티캐스트 그룹 멤버뿐만 아니라 멤버와 멤버를 연결하는 최단 경로 상의 노드들 중 방문 횟수가 가장 높은 노드 또는 이 최단 경로 상의 중간에 위치하여 방문 횟수가 가장 높은 노드를 센터의 후보로 고려한다. 그러므로 멤버만을 고려하는 알고리즘은 물론 제한된 수의 비멤버를 추가로 센터의 후보로 고려하는 기존의 알고리즘보다 좋은 성능을 보이는 센터를 찾을 수 있다. 그리고 알고리즘의 오버헤드도 크지 않으며 이의 수행은 유니캐스트 라우팅 알고리즘에 의존하지 않는 특성을 보인다.

ABSTRACT

Multicast routing algorithms such PIM, CBT, BGMP use shared multicast routing trees and the location of the multicast tree has great impact on the packet delay. In this paper we propose three new center location algorithms and analyze their performance through simulation studies. These three algorithms consider as candidates for the center not only multicast group members but also a few non-member nodes. To select these non-member nodes, we first find all the shortest paths among every couple of members and consider either nodes which are most frequently visited during the process of finding shortest paths or nodes which lie at the center of a shortest path and are most frequently visited during the same process. There the proposed algorithms are able to find the better center than not only algorithms which consider only member nodes but also other algorithms which consider selected non-member nodes in addition to member nodes. The proposed algorithms neither incur too much overhead nor depend upon unicasting algorithms.

I. 서 론

인터넷의 발달과 더불어 부각되고 있는 것이 멀티캐스트 기술이다. 멀티캐스트 기술은 특정 그룹에 데이터를 효율적으로 보내 수 있는 통신방법으로 음성 및 영상 전송과 같은 멀티미디어 응용프로그램을 수행할 수 있도록 한다. 멀티캐스트 방법은 크게 두 가지로 나뉜다. 하나는 송신자에서 모든 수신자까지 최단경로를 갖는 SBT(Source Based Tree) 방식으로 송신자를 중심으로 트리를 만들어 멀티캐스트 데이터를 송신자와 수신자 사이의 최단경로를 통하여 전달하는 방식이다. 또 하나는, 하나의 라우터가 센터가 되고 센터와 모든 멤버간의 최단경로

* 홍익대학교 전자계산학과 분산컴퓨팅 네트워크 연구실 (skkang@cs.hongik.ac.kr)
** 홍익대학교 정보컴퓨터공학부 부교수 (shim@cs.hongik.ac.kr)
논문번호 : 99391-0928, 접수일자 : 1999년 9월 28일
※ 본 연구는 한국과학재단 논문연구과제(KOSEF 981-0917-089-2) 지원으로 수행되었습니다.

를 만든 후 송신자가 데이터를 센터에게 전송하고 센터는 이 데이터를 멤버에게 전송하는 ShT(Shared Tree)방식이다. 이 두 방식을 사용하는 프로토콜을 그림 1에서와 같이 분류할 수 있다[1]. SBT를 이용하는 프로토콜 중에서 DVMRP[2]와 MOSPF[3]는 유니캐스트 라우팅 알고리즘에 밀접하게 의존하여 확장성(scalability)의 문제가 있다. 따라서 대안으로서 ShT를 이용하는 CBT(Core Based Tree)[4]와 SBT와 ShT를 모두 이용하는 PIM(Protocol Independent Multicast)[5]가 제안되었다. 두 알고리즘은 센터(Core, RP)를 이용한 ShT를 사용하여 트리의 생성 및 유지를 위한 오버헤드가 많이 감소하였다. 하지만 인터넷과 같이 여러 개의 도메인으로 구성된 네트워크를 위한 인터도메인 멀티캐스트 라우팅 알고리즘으로 적절하지 않고 하나의 도메인 내에 적용되는 인트라도메인 멀티캐스트 라우팅 알고리즘에 적절하다. 인터도메인 멀티캐스트 라우팅 알고리즘으로는 HDVMRP[6], HPIM[7], BGMP[8] 등이 있다. 이와 같이 인트라도메인, 인터도메인에서 사용하는 프로토콜인 CBT, PIM, BGMP 등은 ShT를 사용하여 멀티캐스트 데이터를 전송하지만, 멀티캐스트 그룹을 처음 생성하는 노드가 센터가 되고 가입과 탈퇴가 많이 일어난 후에도 센터의 위치가 변하지 않는 문제가 있다. VOD나 원격 강의와 같이 그룹 생성자가 주 송신자가 되는 경우에는 문제가 되지 않지만 음성 및 영상 회의와 같이 모든 멤버가 송신자가 되는 경우에는 문제점이 발생할 수 있다. 센터의 위치가 송신자와 수신자와 관계없이 엉뚱한 곳에 위치하면 송신자에서 센터, 센터에서 각 멤버까지의 패킷 지연이 상당히 커지게 되고, 또한 패킷전달을 위한 트리 비용도 높아진다.

위의 문제를 해결하기 위해 센터를 찾기 위한 방

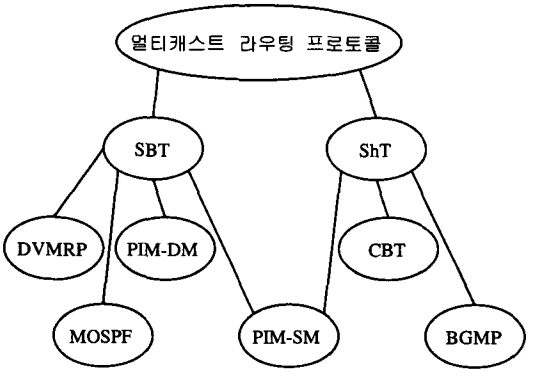


그림 1. 멀티캐스트 라우팅 프로토콜의 분류.

법을 생각해 보면 다음과 같이 나눌 수 있다.

첫째 네트워크의 노드들이 모두 참여를 하여 센터를 구하는 것이다. 즉 멤버들이 위치한 곳을 기준으로 하여 모든 네트워크의 노드들을 조사하여 가장 좋은 노드를 선택하는 것을 말한다. 하지만 이것은 가장 좋은 센터를 얻을 수 있지만 제어 메시지의 전송으로 인한 대역폭(bandwidth)의 많은 소비와 계산에 드는 비용이 크기 때문에 현실적으로 실현되기는 힘들다.

둘째로 멤버만 참여하여 센터를 구하는 것이다. 이 방법은 멤버만을 고려하는 것이므로 모든 노드가 참여하는 첫 번째 방법보다는 제어 메시지의 전송으로 인한 대역폭 소비가 훨씬 적고 계산에 드는 비용이 아주 적게 된다. 하지만 멤버만이 센터가 되기 때문에 멤버가 아닌 더 좋은 노드가 있어도 센터가 될 수 없는 단점이 있다. 왜냐하면 멤버의 수가 아주 적고 넓게 분포한 상황에서, 때로는 멤버가 다른 모든 노드들 보다 좋을 수 있겠지만 멤버보다는 멤버가 아닌 다른 노드가 더 좋을 수 있는 확률이 높기 때문이다. 한 예로 멤버들이 모두 중심 쪽이 아닌 외곽 쪽에 많이 분포하고 외곽에 있는 멤버 중 하나가 센터가 되었다고 한다면 패킷을 전송하였을 때 많은 지연을 유발할 것이다. 따라서 중심 쪽에서 센터가 되기 위한 노드를 찾는 것이 더 좋은 방법이라 생각된다.

셋째는 멤버와 제한된 수의 비멤버들 중 센터를 구하는 것이다. 이 방법은 첫 번째 방법의 확장성 문제와 두 번째 방법의 노드의 제한을 해결한 방법이다. 멤버가 비멤버보다 더 좋은 경우 센터가 되고, 비멤버가 멤버보다 좋은 경우 센터가 되는 것을 말한다. 이 방법은 제한된 비멤버들을 찾는 것이 문제가 된다.

넷째는 현재의 센터의 이웃 노드들 중 하나 또는 이웃의 이웃노드들 중 하나가 센터가 되는 방법이다. 이 방법은 이웃노드가 비멤버이든 멤버이든 상관이 없다. 모든 멤버를 기준으로 이웃노드들의 가중치를 구하여 그 중 센터가 되는 방법이다.

본 논문에서는 세 번째 방법을 이용하여 제한된 수의 비멤버들을 구하게 되고, 구한 비멤버와 멤버 중에서 센터를 선택하는 알고리즘 3개를 제안한다. 이 세 알고리즘은 센터 후보로 고려될 비멤버를 선택하는 방법이 다르다. 각각의 알고리즘은 지리적으로 많이 방문되거나 멤버들의 중심에 위치하는 비멤버노드를 찾는 알고리즘들로 앞으로 각각을 GeoCenter (Geometric/Geographic Center)1,

GeoCenter2, GeoCenter3으로 칭한다. 제한된 수의 비멤버들을 구하기 위해 GeoCenter1과 GeoCenter2에서는 멤버에서 각 멤버까지의 경로 정보를 얻은 후 가장 많이 방문된 노드를 센터의 후보로 고려한다. 여기서 GeoCenter1과 GeoCenter2는 노드의 방문횟수 계산 방식에 따라 나뉜다. GeoCenter3에서는 얻은 경로 정보를 바탕으로 멤버와 멤버들 사이의 중간노드들을 선택하고 이 중 가장 많이 방문된 노드를 센터의 후보로 고려한다. 각각의 알고리즘에서 얻어진 제한된 수의 비멤버가 여럿일 경우 임의로 하나의 노드를 선택하게 되고, 최종적으로 센터는 경로 정보를 통해 선택된 하나의 노드와 멤버들과의 최대 패킷 지연의 값을 비교하여 가장 작은 값을 갖는 노드가 된다.

본 논문에서 제시하는 세 알고리즘은 멀티캐스트 데이터 전달 트리의 비용을 최소화하면서 패킷 지연에서 좋은 성능을 발휘하기 위한 알고리즘이다. 그러나 전달 트리의 비용과 패킷 지연 시간을 최소화하고자 하였다. 그리고 생성된 전달 트리의 비용이 다른 알고리즘들에 의해 생성된 전달 트리와 크게 차이가 나지 않도록 하였다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구의 장단점을 살펴보고 3장에서는 새로운 알고리즘을 제시한다. 4장에서는 새로운 알고리즘의 평가를 위해 실험을 통한 결과를 살펴본다. 5장에서는 새 알고리즘의 오버헤드를 살펴보게 되고 마지막으로 6장에서는 결론을 맺는다.

Ⅱ. 관련 연구

센터를 찾기 위한 여러 알고리즘이 제안되어 있다. 본 장에서는 그 중에서 최근에 제안된 알고리즘을 살펴본다. 먼저 제안된 알고리즘들을 설명하기 전에 트리 비용의 정의와 각 알고리즘에서 사용하는 가중치에 대해 설명한다. 트리 비용이란 하나의 노드가 센터가 되고 멤버들이 이 센터에 가입한 상태 즉 센터가 루트가 되고 멤버가 잎이 되는 트리의 모든 링크 비용의 합이다. 링크의 비용은 실제 네트워크에서는 돈으로 표현하거나 패킷을 전송할 때의 대역폭이나 지연 등으로 나타낼 수 있지만 본 논문에서는 모든 링크의 비용을 1로 가정한다. 가중치(weight)란 센터가 되기 위한 노드들이 갖는 값으로 다른 노드들과 비교하기 위해 사용하는 값을 말한다. 여러 알고리즘에서 사용하는 가중치를 계산하는 공식은 다음과 같으며, 각각의 공식에서 S는 소스 또는 멤버의 집합을, u, v는 S에 속한 소스 또는 멤버를, root는 센터가 되기 위한 노드를 나타낸다[9][10].

1) 실제 트리 비용 함수 (Actual Tree Cost Function)

하나의 노드를 루트로 놓고 모든 멤버까지의 실제 링크 비용 합을 구하는 함수이다.

2) 최대 거리 함수 (Maximum Distance Function)

$$\text{Max Dist} = \max_{u \in S} d(root, u)$$

하나의 노드(root)에서 모든 멤버까지의 거리를 각각 구한 후 그 중 최대 값을 구하는 함수이다.

3) 평균 거리 함수(Average Distance Function)

$$\text{Avg Dist} = \frac{1}{|S|} \sum_{u \in S} d(root, u)$$

하나의 노드(root)에서 멤버까지의 거리를 모두 더한 후 멤버의 수로 나눈 값을 구하는 함수이다.

4) 최대 지름 함수 (Maximum Diameter Function)

$$\text{Max Diam} = \max_{u \in S} d(root, u) + \max_{v \in S, v \neq u} d(root, v)$$

하나의 노드(root)에서 멤버까지의 거리를 각각 구한 후 최대 값과 그 다음 최대 값을 구하여 더하는 함수이다.

5) 추정된 비용 함수 (Estimated Cost Function)

$$\text{Est Cost} = \frac{\text{Est Cost}_{min} + \text{Est Cost}_{max}}{2}$$

여기서 Est Cost_{min}과 Est Cost_{max}는 다음과 같다.

$$\text{Est Cost}_{min} = \max_{u \in S} d(root, u) + \text{number of duplicate distance nodes in S}$$

$$\text{Est Cost}_{max} = \begin{cases} \sum_{u \in S} d(root, u) & \text{if } |S| \leq \deg(root), \\ [\sum_{u \in S} d(root, u)] - [|S| - \deg(root)] & \text{otherwise} \end{cases}$$

트리 비용의 최적과 최악의 경우를 라우팅 테이블을 바탕으로 계산하여 평균을 구하는 함수이다.

트리 비용 측면에서 최적 알고리즘인 OCBT (Optimal Center Based Tree)[9][10]는 네트워크의 각 노드로부터 모든 멤버까지의 최단 거리 트리를 구한 후 트리 비용이 가장 작은 노드를 선택한다. 여러 노드가 선택이 되면 최대 패킷 지연이 가장 작은 것이 센터가 된다. 이 알고리즘은 모든 노드가 참여하여 노드마다 트리 비용을 구해야 하므로 상

당한 오버헤드가 발생한다. 따라서 실제 네트워크 환경에서 사용 불가능하다.

RSST(Random Source Specific Tree)는 PIM과 CBT에서 제안한 것으로 첫 번째 소스(source)나 멀티캐스트 그룹 생성자(initiator)가 센터가 되는 방식이다. 이 알고리즘은 센터가 하나만 존재하게 되고 비용이나 패킷 지연이 크지만 한번 센터가 정해지면 바뀌지 않으므로 센터가 바뀔 때 생기는 비용은 없다.

MIN-MEMB(Minimal Member Tree)[9][10]은 현재의 센터가 자신의 가중치(weight)를 구한 후 모든 멤버에게 멤버리스트와 함께 전송한다. 여기서 가중치는 위에서 살펴본 다섯 가지의 공식 중 하나를 사용하여 구하게 된다. 센터가 되기 위한 노드에서는 자신의 가중치를 구하고 일정시간 기다린 후 다른 멤버에게 멀티캐스트하게 된다. 이렇게 일정시간을 기다리는 이유는 모든 센터가 되기 위한 노드가 자신의 가중치를 멀티캐스트하면 제어 패킷의 범람을 초래할 수 있기 때문이다. 다른 센터가 되기 위한 노드에서는 자신의 가중치를 현재 받은 값들과 비교하여 더 좋을 경우 멀티캐스트하게 된다. 현재의 센터에서는 멀티캐스트되어진 가중치들을 모아 후보 리스트를 만들고 가중치가 가장 낮은 노드를 선택하여 멤버/소스 리스트를 전달하여 센터가 되게 하는 방식이다. 가중치 공식 5인 추정된 비용함수를 사용하고, 그룹 멤버 리스트 또는 소스 리스트 중 어떤 것을 사용하느냐에 따라 MEMMT (Minimum Estimated Member-Member Tree) 또는 MEMST (Minimum Estimated Member -Source Tree)로 불린다. 이 방식은 위에서 언급한 두 번째 방식인 멤버만이 센터가 되는 단점이 있다.

HILLCLIMB(Hill-Climb Protocol)[9][10]은 센터를 다음과 같이 구하게 된다. 우선 이 알고리즘에는 3가지의 리스트가 필요하다. 첫째 경로 리스트(path list)이다. 이 리스트는 센터가 되기 위한 노드들이 지나온 순서대로 기록된다. 따라서 리스트의 첫 번째는 현재의 센터가 기록되게 된다. 그 다음으로 다음에 조사한 노드들을 기록해나가는 것이다. 따라서 이 리스트의 크기를 제한하여 알고리즘이 끝나는 것을 명시할 수 있다. 예를 들어 크기를 3으로 두면 센터가 되기 위한 노드를 조사하는 과정에서 현재의 센터가 리스트의 첫 번째에 기록되고 현재의 센터보다 좋은 가중치를 갖는 다음 조사 노드가 두 번째에 기록되고 두 번째 보다 좋은 가중치를 갖는 다음 조사 노드가 세 번째에 위치하게 된다. 크기를 3으로 두었으므로 세 번째가 센터가 되는 것이다. 만약 두 번째 보다 좋은 노드가 없으면 두 번째가 센터가 된다. 두 번째 리스트는 방문 리스트(visited/unvisited list)이다. 이 리스트는 다음 조사할 노드를 선택할 때 사용한다. 즉 방문한 노드는 제외하고 방문하지 않은 노드 중에서 가중치가 가장 낮은 노드를 선택하여 다음 조사에 사용하게 된다. 세 번째는 n개의 최적의 후보 리스트이다. 이 리스트는 다른 알고리즘에도 있는 것으로 현재의 센터가 문제가 발생하여 센터의 역할을 할 수 없을 경우 이 리스트에서 다음 센터를 선택할 때 사용한다. 따라서 후보가 될 수 있는 노드들을 조사할 때 계속해서 수정해야한다.

HILLCLIMB 알고리즘에서는 센터를 구하기 위해 현재의 센터가 이웃 노드들(neighbors)에게 가중치를 물어보면 이웃노드들은 자신의 가중치를 구한 후 현재의 센터에게 보내게 된다(가중치를 구하기 위해 위의 가중치 공식 중 하나를 사용한다.). 현재의 센터는 자신의 가중치를 구한 후 이웃 노드들의 가중치와 비교하여 값이 작은 이웃 노드들을 n개의 최적의 후보 리스트에 등록을 하고 또한 방문 리스트에도 기록하게 된다. 만약 이웃노드들의 가중치보다 현재의 센터의 가중치가 작다면 현재 센터가 다시 센터가 된다. 그러나 그렇지 않다면 방문 리스트 중에서 가장 가중치가 작은 노드를 선택하여 경로 리스트와 멤버/소스리스트를 전달하여 다음 조사를 하게 된다. 경로 리스트와 멤버/소스리스트를 전달받은 노드에서는 다시 이웃노드들에게 가중치를 물어본다. 이웃노드들은 자신의 가중치를 구한 후 현재 조사하는 노드에게 전달하고 현재 조사하는 노드는 자신의 가중치보다 작은 노드를 방문노드에 기록한다. 이 때 자신보다 가중치가 낮은 노드가 없으면 지금 조사하는 노드가 센터가 되고 그렇지 않으면 조사하는 노드는 경로리스트에 추가하고 다시 방문리스트에서 다음노드를 선택한 후 계속조사를 한다. 이런 과정을 반복하여 가중치가 낮은 노드들을 계속 방문하고 계속 조사를 하여 센터가 되기 위한 노드를 찾아 센터가 되는 방식이다. 가중치 공식 5인 추정된 비용 함수를 사용하고, 소스리스트를 사용하면 HC-S로 불리고 멤버리스트를 사용하면 HC-M으로 불린다. 이 알고리즘은 경로 리스트의 크기를 정하는 것이 문제이다. 경로리스트 크기를 작게 주면 알고리즘은 빨리 끝날 수 있지만 현재의 센터보다 훨씬 좋은 성능을 갖는 센터를 구할 수 없고 크게 주면 알고리즘이 늦게 끝나게 된다. 그리

고 찾은 센터가 지역적으로 최적일 경우 더 좋은 센터가 있어도 찾지 못하는 단점이 있다.

MIN-MEMB와 HILLCLIMB은 가중치 공식 5를 사용할 경우 실제 거리정보를 이용하여 가중치를 구하는 것이 아니라 라우팅 테이블에 존재하는 거리 정보를 바탕으로 구하는 방식이다. 따라서 영역(domain)이 다를 경우 거리를 알 수 없게 되어 큰 네트워크 환경에서는 사용 불가능하다.

MCT(Maximum-Centered Tree), ACT(Average -Centered Tree), DCT(Diameter-Centered Tree)들은 모든 네트워크 노드가 참여하게 되며 각각의 노드는 MCT의 경우 가중치 공식 2, ACT의 경우 3, DCT의 경우 4를 사용하여 자신의 가중치를 구한다. 세 알고리즘은 각각의 노드에서 구한 가중치중 가장 작은 값을 갖는 노드가 센터가 되는 방식이다. 실제 모든 네트워크의 노드가 참여하므로 실제 상황에 적용하기는 어렵다.

Ⅲ. 센터 위치 결정을 위한 새 알고리즘

본 장에서는 공유된 멀티캐스트 트리의 센터 위치 결정을 위한 새 알고리즘들인 GeoCenter1, GeoCenter2, GeoCenter3에 대해 설명한다. 이 알고리즘들은 모두 멤버들 사이의 경로 정보를 바탕으로 센터를 구한다. 이 경로 정보를 얻기 위해 traceroute를 이용하거나 IP헤더에 기록하는 방법들을 사용하므로 유니캐스트 라우팅 알고리즘에 독립적으로 센터를 찾을 수 있다.

GeoCenter1 알고리즘에서는 각 멤버가 다른 모든 멤버들까지의 경로정보를 구하여 이 경로 상에 놓이는 비멤버 노드의 리스트를 구한 후 이 리스트를 센터에 보낸다 센터는 이 리스트들 중에서 나타나는 횟수가 가장 많은 노드를 찾는다 만약 여러 노드가 찾아지면 이 중 하나를 임의로 선택한다. 이렇게 선택된 비멤버 노드와 모든 멤버 노드들이 센터의 후보로 고려된다. 이들 후보들 중 최대 거리 합수가 가장 작은 노드가 센터가 된다.

GeoCenter2와 GeoCenter3 알고리즘은 후보로 고려될 비멤버 노드를 선택하는 방법면에서 GeoCenter1과 다르다. GeoCenter2 알고리즘에서 각 멤버는 다른 멤버들까지의 경로를 구한 후 이 경로들 상에 나타나는 노드들의 리스트뿐 아니라 이 노드들이 이 경로들을 구하는 과정 중 몇 번 방문되었는지의 횟수도 함께 센터에 보낸다. 센터는 모든 리스트 중에서 방문된 횟수가 가장 많은 노드를 비멤버 센터 후보로 고려한다. GeoCenter3 알고리즘에서 각 멤버는 다른 멤버들까지의 경로를 구한 후 이 경로 상의 중간 위치에 놓이는 노드들의 리스트를 센터에 보내고 센터는 이 중 가장 많이 방문된 노드를 비멤버 센터 후보로 고려한다.

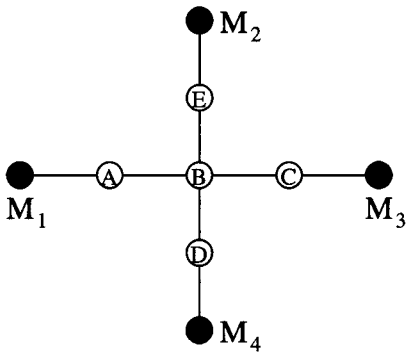


그림 2. 기본 예.

제안된 알고리즘들은 간단한 예를 사용하여 설명한다. 그림 2에서 M1, M2, M3, M4는 멤버이고, A, B, C, D, E는 이들 멤버들 간의 최단 경로에 놓이는 비멤버 노드들이다. 즉 M1과 M2 사이에는 A, B, E가 M1, M3사이에는 A, B, C가 M1과 M4사이에는 A, B, D가 위치한다. GeoCenter1 알고리즘에서 각 멤버는 다음과 같은 정보를 센터에 보내고

M1 : A, B, C, D, E
M2 : A, B, C, D, E
M3 : A, B, C, D, E
M4 : A, B, C, D, E

센터는 이들 리스트들에 나타난 노드들의 횟수가 4로 전부 같으므로 이 중 임의의 노드를 센터의 후보로 고려한다. GeoCenter2 알고리즘에서 각 멤버는 다음과 같은 정보를 센터에 보낸다.

M1 : (A, 3), (B, 3), (C, 1), (D, 1), (E, 1)
M2 : (A, 1), (B, 3), (C, 1), (D, 1), (E, 3)
M3 : (A, 1), (B, 3), (C, 3), (D, 1), (E, 1)
M4 : (A, 1), (B, 3), (C, 1), (D, 3), (E, 1)

이 리스트들에 나타난 방문 횟수를 전부 더하게 되면

(A, 6), (B, 12), (C, 6), (D, 6), (E, 6)

의 결과가 나오므로 센터는 이 중 방문 횟수가 가

장 높은 B를 비멤버 센터 후보로 고려한다. GeoCenter2에서는 멤버들 사이의 경로가 겹치는 노드가 더 많은 방문 횟수를 가지게 되므로 GeoCenter1 알고리즘보다 더 좋은 노드가 후보로 고려될 수 있다. GeoCenter3 알고리즘에서 각 멤버는 다음과 같은 정보를 센터에 보낸다.

M1 : (B, 3)
M2 : (B, 3)
M3 : (B, 3)
M4 : (B, 3)

그러므로 센터는 B를 비멤버 센터 후보로 고려한다. GeoCenter1 알고리즘과 GeoCenter2 알고리즘은 경로정보를 전달하기 위해 많은 양의 데이터를 전달하여야 하나 GeoCenter3 알고리즘에서는 중간 노드만을 전달하면 되므로 적은 양의 데이터만을 전달하면 된다.

기본적인 아이디어를 바탕으로 본 논문에서 제안하는 세 알고리즘을 자세히 살펴보면 다음과 같다.

1. GeoCenter1:

① 멀티캐스트 그룹이 만들어지게 되면 생성자 또는 송신자가 임시의 센터가 된다.
② 센터는 각 멤버에게 멤버리스트가 포함된 시험 메시지(probe message)를 전달한다.
③ 시험 메시지를 받은 멤버들은 각 멤버들의 경로를 추적하여 경로 정보를 얻게 된다(traceroute 사용 또는 IP헤더에 기록). 이 때 각각의 멤버는 모든 멤버들간의 패킷 지연 값 중에서 가장 큰 값을 자신의 가중치로 한다.
④ 각 멤버들은 방문한 라우터들을 기록하여 센터로 보내고, 자신의 가중치도 센터로 보낸다.
⑤ 현재의 센터에서는 각 노드에서 보낸 경로 정보를 바탕으로 라우터들의 방문 횟수가 높은 노드(들)를 선택하게 된다. 만약 방문 횟수가 가장 높은 노드가 여럿일 경우 임의의 하나만을 선택한다. 선택된 노드가 멤버일 경우 ⑧을 수행한다
⑥ 선택된 멤버가 아닌 노드(들)에게 멤버 리스트와 함께 시험 메시지를 보낸다.
⑦ 선택된 노드(들)는 멤버 리스트와 시험메시지를 받으면 모든 멤버에 'ping'을 보내어 가장 큰 패킷 지연 값을 자신의 가중치로 하여 센터로 전송하게 된다.
⑧ 센터에서는 선택된 노드와 모든 멤버의 가중치를 비교하여 값이 가장 작은 노드를 센터로 정한다.
⑨ 현재의 센터는 모든 멤버에게 멀티캐스트 하여 새로운 센터를 알리게 되고, 새롭게 선택된 노드에게 그룹 멤버 리스트와 후보 센터리스트를 보내게 된다.

2. GeoCenter2 :

GeoCenter2 알고리즘은 GeoCenter1 알고리즘과 동일하지만 ④번에서 각각의 멤버는 라우터의 방문 횟수를 계산하게된다.

3. GeoCenter3 :

①②③ GeoCenter1 알고리즘과 동일하다.
④ 각 멤버들은 방문한 라우터들의 정보를 바탕으로 멤버와 멤버사이의 중간 라우터를 구하게된다. 중간 라우터를 모두 구하게 되면 자신의 가중치와 선택된 중간라우터의 방문횟수를 센터에게 전달한다.
⑤ 현재의 센터에서는 중간라우터의 방문 횟수를 모두 더하여 가장 큰 값을 갖는 노드(들)를 선택한다. 만약 가장 큰 값을 갖는 노드가 멤버일 경우 ⑧을 수행한다.
⑥⑦⑧⑨ GeoCenter1 알고리즘과 동일하다.

Ⅳ. 실험 결과

본 장에서는 이 세 알고리즘들이 선택한 센터를 바탕으로 트리를 구성하였을 때, 그룹 크기와 네트워크 크기, 평균 링크의 수의 변화에 따른 트리 비용과 패킷 전달 지연에 대하여 다른 알고리즘들과 성능비교를 한다. 성능 비교를 위해 사용한 알고리즘은 트리 비용에서 최적인 OCBT, 패킷 지연에서 거의 최적인 MDOT(Minimum Delay Optimal Tree), MIN-MEMB, HILLCLIMB, 그리고 멤버 중에 임의로 선택하는 Random 방식이다. 패킷 지연에 거의 최적인 MDOT는 실험을 위하여 생각한 알고리즘으로 네트워크의 각 노드를 센터로 고려하였을 때 최대 지연 함수 값이 가장 작은 노드를 센터로 고려하는 방법으로 네트워크의 모든 노드가 참여하게 된다. 만약 같은 값을 갖는 노드가 여러 개일 경우 트리 비용이 작은 것인 센터가 된다. 이 알고리즘은 PIM과 같이 단방향 방식에서의 최적의 알고리즘이고, OCBT와 마찬가지로 제안된 알고리즘들과 비교하기 위하여 생각한 것으로 실제 네트

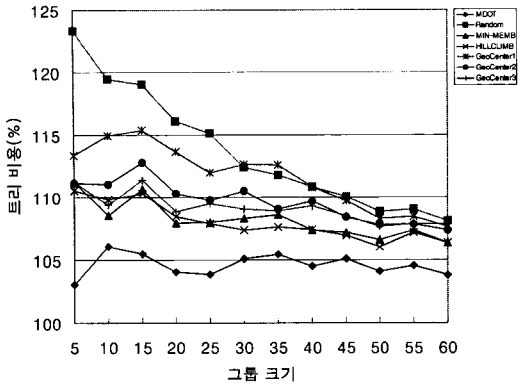


그림 3. 그룹 크기 변화에 따른 트리 비용의 비교

워크에 적용할 수 없다.

본 실험은 NS(Network Simulator)[11]를 사용하였고, OS는 Linux5.1를 사용하였다. 그리고 각각의 알고리즘은 TCL을 사용하여 구현하였으며, 임의의 네트워크 토폴로지를 얻기 위해 NS에서 제공하는 GT-ITM(Georgia Tech Internetwork Topology Models)[12]을 사용하였다.

본 실험에서 사용한 파라미터로는 네트워크 크기, 그룹 크기, 평균링크의 수(degree)이고 송신자는 항상 멤버라고 가정하였고, 각 실험은 100번씩 수행한 후 평균을 구하였다.

1. 그룹 크기 변화에 따른 비교

그룹의 크기란 전체 네트워크에서 얼마만큼의 노드가 멀티캐스트에 참여하는가를 나타내는 것이다. 즉 멀티캐스트 데이터를 받는 수신자의 수가 늘어난다는 것을 의미한다. 본 실험에서는 전체 네트워크의 크기가 100으로 보았을 때 그룹의 크기를 변화하여 트리 비용과 패킷 지연을 살펴본다.

1.1 트리 비용

그림 3은 그룹 크기 변화에 따른 트리 비용을 OCBT와의 비율로서 나타내었다. MDOT가 OCBT 다음으로 좋은 것으로 나타났다. MIN-MEMB와 HILLCLIMB는 비슷한 성능을 보였으며 약간의 차이는 있었지만 GeoCenter3, GeoCenter2와 비슷한 성능을 보였다. GeoCenter3의 경우 그룹의 크기가 커짐에 따라 Random과 비슷한 성능을 보였다.

1.2 패킷 지연

패킷 지연을 구하는 방법은 멀티캐스트 트리가 PIM과 같이 단방향 트리인 경우와 CBT나 BGMP와 같이 양방향 트리인 경우 각각 다르게 된다. 단방향 트리에서 패킷은 먼저 센터로 전달된 후 멤버들에게 전달되게 된다. 반면 양방향 트리에서 패킷은 공유된 멀티캐스트 트리상에서 송신자와 수신자 사이의 최단 경로를 따라 전달되게 되고, 센터를 지나지 않는 경우도 있게 된다. 따라서 단방향의 경우 송신자에서 센터까지의 최단거리와 센터에서 멤버까지의 최단거리의 합 중에서 가장 큰 값을 각 알고리즘의 비교의 기준으로 한다. 양방향일 경우 송신자에서 멤버까지 센터를 경유하는 것과 경유하지 않는 경우를 모두 고려하여 가장 큰 값을 비교의 기준으로 한다.

그림 4는 단방향 트리에서의 패킷 지연을 보인 것으로 GeoCenter2, GeoCenter3이 MIN-MEMB와 HILLCLIMB보다 좋은 성능을 나타내었다. GeoCenter1의 경우 그룹 크기가 20, 40인 경우를 제외하고 모두 좋은 것으로 나타났다. 그룹 크기가 35에서 GeoCenter3이 MDOT보다 좋은 것으로 나타난 것은 MDOT가 송신자를 고려하지 않고 멤버들만을 고려하여 센터를 만들었기 때문이다. 즉 송신자에서 센터까지의 거리와 센터에서 멤버까지의 거리의 합을 고려하지 않고, 단지 멤버까지의 거리만을 고려하였기 때문이다. 따라서 이러한 경우가 그룹 크기 35에서 많이 발생하였다.

그림 5에서도 그림 4와 마찬가지로 GeoCenter2, GeoCenter3이 MIN-MEMB과 HILLCLIMB보다 좋은 성능을 나타내었고, GeoCenter1의 경우 MIN-MEMB과 HILLCLIMB과 비슷한 성능을 나타내었다. 그림 5에서 GeoCenter2, GeoCenter3알고리즘이 MDOT 보다 좋은 경우가 있는 것은 MDOT는 단방향 방식에서 최적이기 때문이다.

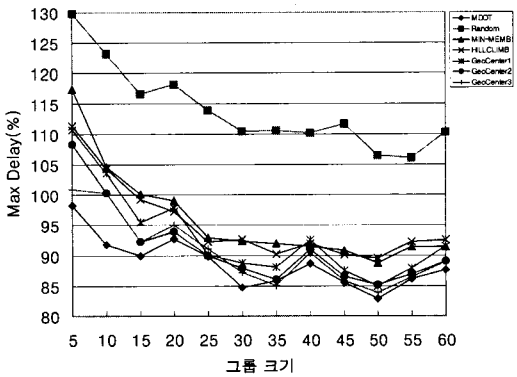


그림 4. 그룹 크기 변화에 따른 패킷 지연의 비교 (단방향 트리)

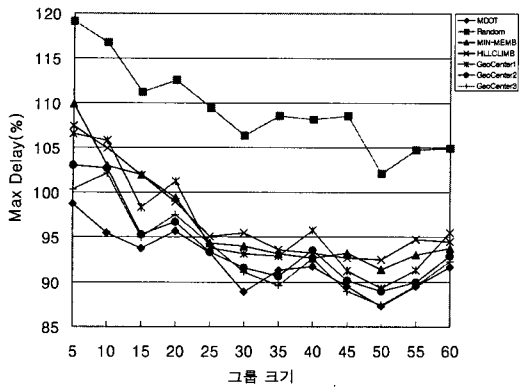


그림 5. 그룹 크기 변화에 따른 패킷 지연의 비교 (양방향 트리)

2. 네트워크 크기 변화에 따른 비교

네트워크 크기는 네트워크 내의 총 노드의 수로 생각할 수 있다. 따라서 네트워크의 크기 변화에 따른 비교를 해봄으로써 큰 네트워크에서 어떻게 동작하는지 성능 비교를 해 볼 수 있다. 네트워크의 크기는 50에서 300까지 변화를 주었고, 평균 링크 수는 4로 주었다. 그리고 그룹의 크기는 네트워크 크기의 20%로 하였다.

2.1 트리 비용

그림 6은 네트워크 크기 변화에 따른 트리 비용의 비교를 나타내는데 네트워크의 크기가 200에서 300일 경우 GeoCenter2, GeoCenter3이 MIN-MEMB와 HILLCLIMB과 비슷한 성능을 보였다.

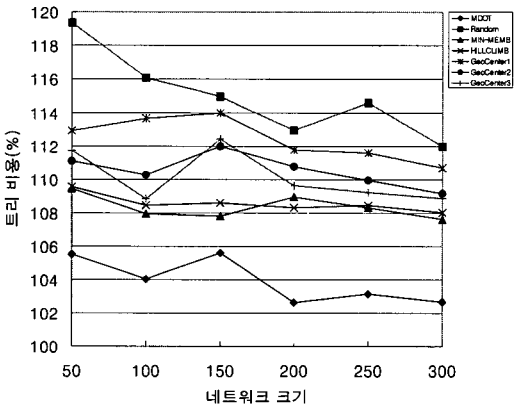


그림 6. 네트워크 크기 변화에 따른 트리 비용의 비교

2.2 패킷 지연

그림 7에서 GeoCenter2, GeoCenter3이 MIN-MEMB와 HILLCLIMB보다 좋은 성능을 나타낸다. 또한 MDOT보다 좋을 때도 있었다. GeoCenter1의 경우 MIN-MEMB과 HILLCLIMB와 비슷한 성능을 보였다.

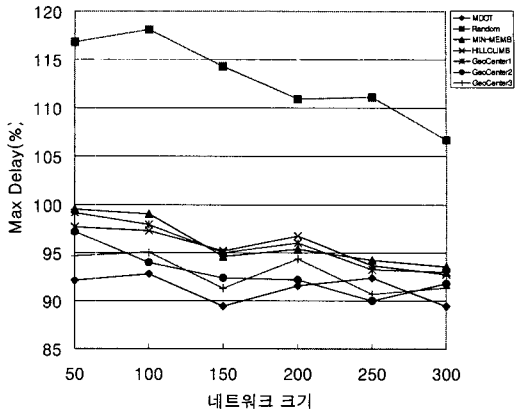


그림 7. 네트워크 크기 변화에 따른 패킷 지연의 비교 (단방향 트리)

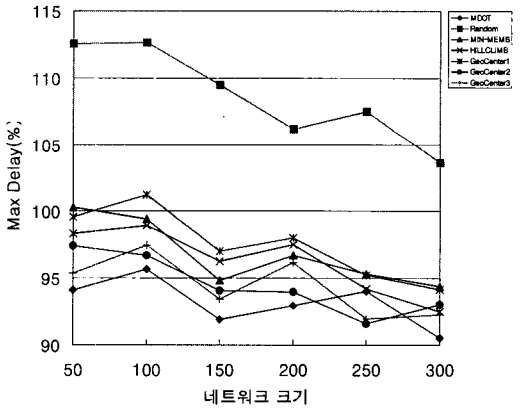


그림 8. 네트워크 크기 변화에 따른 패킷 지연의 비교 (양방향 트리)

양방향 트리일 경우 그림 8과 같다. GeoCenter2, GeoCenter3가 MIN-MEMB과 HILLCLIMB 보다 좋고 GeoCenter1의 경우 MIN-MEMB과 HILLCLIMB와 비슷하였다.

3. 평균 링크 수의 변화에 따른 비교

네트워크의 크기는 100으로 하고 그룹 크기는 네트워크의 크기에 20%인 20으로 하여 노드들의 평균 링크 수의 변화에 따른 각 알고리즘의 트리 비용과 패킷 지연을 비교하였다. 동일한 네트워크 크기에서 평균 링크 수의 커진다는 것은 그 만큼 하

나의 노드에서 하나의 노드로의 연결될 확률이 높아진다는 것이므로 멤버와 멤버간의 거리가 짧아진다는 것을 의미한다.

3.1 트리 비용

그림 9에서 평균 링크 수의 변화에 따라 MDOT가 OCBT 다음으로 좋은 성능을 보였다. 그리고 HIILLCLIMB와 MIN-MEMB가 거의 비슷한 성능을 보여주었다. 알고리즘 GeoCenter1, GeoCenter2, GeoCenter3은 HIILLCLIMB와 MIN-MEMB보다는 최고 6%정도 뒤졌다. 하지만 GeoCenter3은 평균 링크 수가 3일 경우 HIILLCLIMB와 MIN-MEMB보다 좋았다.

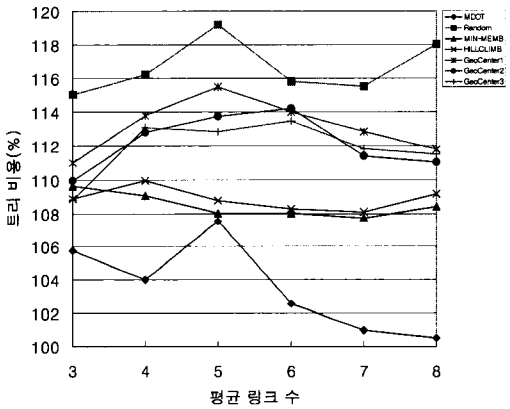


그림 9. 평균 링크 수 변화에 따른 트리 비용의 비교

3.2 패킷 지연

그림 10과 같이 단방향 트리에서 GeoCenter1, GeoCenter2, GeoCenter3은 평균링크 수가 3에서 6일 때 HILLCLIMB과 MIN-MEMB보다 좋은 성능을 보였으며 7, 8에서는 Random과 OCBT를 제외한 모든 알고리즘이 거의 비슷한 결과를 보여주었다. 그림 11과 같이 양방향 트리에서는 GeoCenter1은 평균링크 수가 3에서 6일 경우 MIN-MEMB보다 좋고 HILLCLIMB와 비슷하였지만 GeoCenter2와 GeoCenter3은 더 좋은 결과를 보여주었다. 그림 10과 비슷하게 평균 링크 수가 7과 8일 경우 Random과 OCBT를 제외한 모든 알고리즘이 비슷한 결과를 보였다.

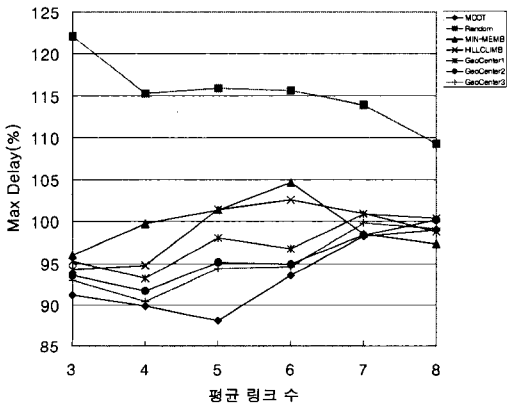


그림 10. 평균 링크 수 변화에 따른 패킷 지연의 비교 (단방향 트리)

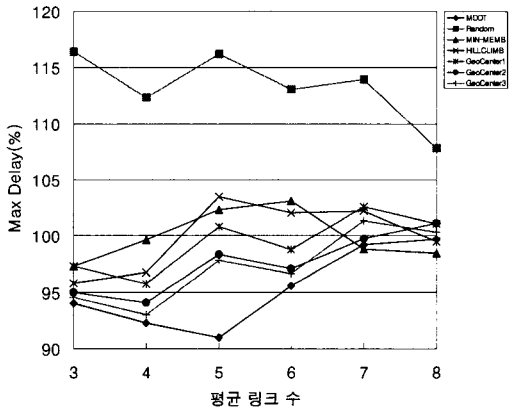


그림 11. 평균 링크 수 변화에 따른 패킷 지연의 비교 (양방향 트리)

V. 오버헤드

본 장에서는 GeoCenter1, GeoCenter2, GeoCenter3 알고리즘이 후보 센터 리스트를 선택하기 위해 사용하는 제어 메시지의 전송 횟수와 전체 네트워크에서 참여하는 노드의 수를 살펴본다. 표1은 각 알고리즘 제어 메시지 전송 횟수와 참여노드 수를 각 알고리즘 별로 비교한 것이다. 표1에서 N은 네트워크의 크기를 나타내고 M은 그룹의 크기를 말하며, H는 홉 수, D는 한 노드의 평균 링크 수를 말한다.

OCBT의 제어 메시지의 전송은 하나의 멤버가 모든 노드에게 한번씩 전송을 해야 함으로 O(N · M)의 메시지 전송이 필요하다. 참여하는 노드의 수는 모든 네트워크 참여함으로 O(N)이 된다.

MIN-MEMB와 HILLCLIMB은 가중치 공식 5를 사용하였다고 가정하여 오버헤드를 생각하였다. 이것은 라우팅 테이블에 거리정보를 구하기에 충분한 정보가 있다고 생각하였지만 실제 다른 도메인일 경우 거리의 정보가 불충분하여 메시지의 전송 횟수는 많이 늘어날 것이다. MIN-MEMB의 센터리스

표 1. 각 알고리즘의 오버헤드 비교

알고리즘	제어 메시지 전송 횟수	참여 노드 수
OCBT	O(MN)	O(N)
MIN-MEMB	$O(M^2)$	O(M)
HILLCLIMB	O(HD)	O(HD)
GeoCenter1	$O(M^2)$	O(M)
GeoCenter2	$O(M^2)$	O(M)
GeoCenter3	$O(M^2)$	O(M)

트를 구하기 위한 제어 메시지 전송 횟수는 최악의 경우 M-1번의 멀티캐스트를 모든 멤버에게 해야한다. 비교의 조건을 맞추기 위해 유니캐스트로 생각하면 $O(M^2)$이 된다. HILLCLIMB의 제어 메시지 전송 횟수는 총 (2D + 1)H이다. 즉 자기의 이웃에게 멤버/소스리스트를 전송한 후 응답을 받는 것이 2D이고 선택된 다음 후보 노드에 경로리스트와 멤버/소스리스트를 전달하는 것이 1번이다. 이것을 몇 번의 홉을 거치는 가를 나타내는 것이 H이다. (여기서 D는 노드의 링크 수(degree)를 말하고 H는 센터를 찾기 위해 몇 번의 홉(Hop)을 거쳤는지를 나타낸다.) 따라서 O(H · D)만큼의 메시지가 전송된다. 참여하는 노드의 수는 가중치 공식 5인 추정된 비용 공식을 사용한다면 멤버는 참여하지 않고 이웃 노드의 수에 홉 수를 곱한 값이 된다. 따라서 O(H · D)가 된다.

GeoCenter1, GeoCenter2, GeoCenter3은 모두 경로 정보를 이용한다. 경로 정보를 얻기 위하여 각 멤버가 모든 다른 멤버들 제어 메시지를 전송함으로 M(M-1)만큼의 메시지 전송이 필요하다. 센터를 구하기 위해 필요한 제어 메시지의 전송 횟수는 $O(M^2)$으로 나타낼 수 있다. 여기서 제한된 수의 노드들 중 하나의 노드를 선택하고 이 노드가 모든 멤버에게 ping을 보내는 메시지의 수는 O(M)이기 때문에 포함되지 않는다. 참여하는 노드의 수는 멤버와 선택된 하나의 후보로 O(M)으로 나타낼 수 있다.

Ⅵ. 결 론

급속한 인터넷의 발달과 더불어 멀티캐스트의 필요성이 더욱 부각되고 있다. 효율적으로 데이터를 전송하기 위해 멀티캐스트 라우팅 알고리즘이 많이 제안되어 왔고 그 중 공유 트리를 통한 패킷의 전달방식이 확장성 문제를 해결하는 방법으로 알려지고 있다. 하지만 최근 제안된 PIM, CBT, BGMP 등과 같은 멀티캐스트 라우팅 알고리즘들은 멀티캐스트 그룹을 처음 만들 때의 생성자 또는 송신자가 센터가 되어 패킷 지연과 패킷 전달 비용이 커지는 문제를 갖게된다. 이에 따른 여러 방법들이 제안되어 왔다. 본 논문에서는 멤버들이 직접 멤버들 간의 경로를 추적하여 방문한 횟수가 많은 노드 또는 멤버와 멤버의 중간에 위치하는 노드를 선택하여 이 노드를 멤버와 함께 센터의 후보로 고려하는 알고리즘을 제안하였다. 본 논문에서 제시한 세 개의 알고리즘은 트리 비용 측면에서 최소화하고 패킷 지연 측면에서 좋은 성능을 발휘하도록 고려하였고, 시뮬레이션을 통하여 제안된 알고리즘들을 기존의 센터 결정 알고리즘들과 비교한 결과 트리 비용에서는 비슷한 성능을 보였지만 패킷 지연에서는 좋은 성능을 보였다. 또 제한된 알고리즘은 센터를 구하는데 필요한 오버헤드가 그리 크지 않고 유니캐스팅 알고리즘에 의존하지 않는 특성을 지니고 있다.

참 고 문 헌

[1] T. A. Maufer, *Deploying IP Multicast in the Enterprise*, Prentice Hall, December 1997.

[2] D. Waitzman, C. Partridge, S. Deering, "Distance Vector Multicast Routing Protocol," RFC 1075, November 1988.

[3] J. Moy, "MOSPF: Analysis and Experience," RFC 1585, March 1994.

[4] B. Cain, Z. Zhang, and A. Ballardi. "Core Based Trees (v3) Multicast Routing: Protocol Specification," Internet Draft draft-ietf-idmr-cbt-spec-v3-00.txt, August 1998.

[5] D. Estrin, M. Handley, A. Helmey, S. Deering, D. Farinacci, M. Handely, V. Jacobson, C. Liu, P. Sharma, and L. Wei, "Protocol Independent Multicast-Sparse Mode (v2): Protocol Specification," RFC 2362, August 1998.

[6] A. Thyagarajan and S. Deering, "Hierarchical Distance-Vector Multicast Routing Protocol," *ACM SIGCOMM'95 Conference,* pp.60-66, September 1995.

[7] M. Handley, J. Crowcroft, I. Wakeman,

"Hierarchical Protocol Independent Multicast (HPIM)," University of College London, ftp://cs.ucl.ac.uk/darpa/hpim.ps, October 1995.

[8] S. Kurmar, D. Thaler, C. Alaettinoglu, D. Estirin, M. Handley, "The MASC/BGMP Architecture for Inter-Domain Multicast Routing," *ACM SIGCOMM Conference Computer Communication Review*, pp.93-103, August 1998.

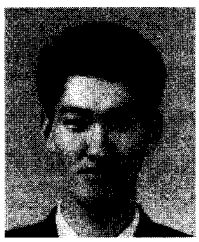
[9] D. Thaler and C. Ravishankar, "Distributed Center-Location Algorithms: Proposals and Comparisons," *IEEE Infocom'96 The Conference on Computer Communication*, 1c.2.1-1c.2.10, March 1996.

[10] D. Thaler and C. Ravishankar, "Distributed Center-Location Algorithms," *IEEE Journal on Selected Area in Communications*, 15(3) pp.281-303, April 1997.

[11] UCB/LBNL/VINT, "Network Simulator-ns (v2)," http://www-mash.cs.berkeley.edu/ns/, 1999.

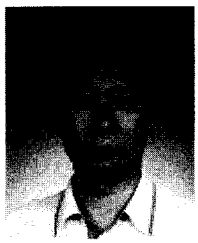
[12] Ellen W. Zegura, "Modeling Topology of Large Internetworks," http://www.cc.gatech.edu/fac/Ellen.Zegura/graphs.html, 1999.

강 신 규(Shin-kyu Kang) 준회원



1998년 2월 : 한성대학교 컴퓨터공학과 학사
1998년 9월~현재 : 홍익대학교 전자계산학과 석사과정
<주관심 분야> 멀티캐스트, 라우팅, 컴퓨터와 망 보안

심 영 철(Young-chul Shim) 정회원



1979년 2월 : 서울대학교 전자공학과 학사
1981년 2월 : 한국과학기술원 전기 및 전자과 석사
1991년 12월 : University of California, Berkeley, 전산학 박사
1981년~1984년 : 삼성전자 대리
1991년~1993년 : University of California, Berkeley, 연구원
1993년~현재 : 홍익대학교 정보컴퓨터공학부 부교수
<주관심 분야> 분산병렬처리, 인터넷 프로토콜, 컴퓨터와 망 보안, 망관리