(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2012-0118382
(43) 공개일자 2012년10월26일

(51) 국제특허분류(Int. Cl.)
H04L 12/28 (2006.01) *H04W 84/18* (2009.01)
(21) 출원번호 10-2011-0035905
(22) 출원일자 2011년04월18일
심사청구일자 2012년06월04일
기술이전 희망 : 기술양도

(71) 출원인
울산대학교 산학협력단
울산 남구 무거2동 산29
(72) 발명자
오훈
서울특별시 양천구 신정로7길 60-7, 푸른마을아파트 406동 402호 (신정동)
한충딘
울산광역시 남구 대학로21번길 9, 정골 4길 11 102호 (무거동)
(74) 대리인
김종선

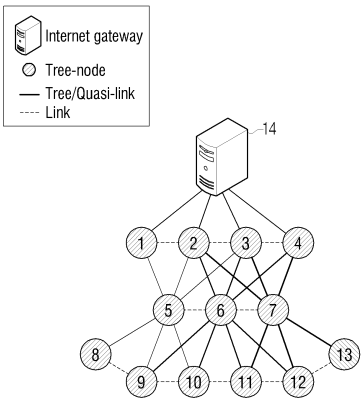
전체 청구항 수 : 총 7 항

(54) 발명의 명칭 **이동성 애드 혹 네트워크에서의 트리 형성 방법**

(57) 요 약

트리 형성 방법이 개시된다. 이동성 애드 혹 네트워크(Mobile ad hoc networks : MANETs)에서의 트리 형성 방법은, 인터넷 게이트웨이(internet gateway)에 연결된 복수의 노드 및 복수의 노드 각각에 순차적으로 연결된 적어도 하나의 노드를 마련하는 단계, 적어도 하나의 노드 중 제1 노드가 자신의 부모 노드로 유니캐스트한 메시지를, 제1 노드의 인접 노드인 제2 노드가 수신하는 단계, 제2 노드가 수신된 메시지에 기초하여 자신의 부모 노드와 인터넷 게이트웨이 사이의 제1 링크 수 및 제1 노드와 인터넷 게이트웨이 사이의 제2 링크 수를 비교하는 단계 및 비교 결과에 따라 제2 노드가 제1 노드를 자신의 예비 부모 노드로 추가 또는 자신의 부모 노드로 설정하는 단계를 포함한다.

대 표 도 - 도2



특허청구의 범위

청구항 1

이동성 애드 혹 네트워크(Mobile ad hoc networks : MANETs)에서의 트리 형성 방법에 있어서,

인터넷 게이트웨이(internet gateway)에 연결된 복수의 노드 및 상기 복수의 노드 각각에 순차적으로 연결된 적어도 하나의 노드를 마련하는 단계;

상기 적어도 하나의 노드 중 제1 노드가 자신의 부모 노드로 유니캐스트한 메시지를, 상기 제1 노드의 인접 노드인 제2 노드가 수신하는 단계;

상기 제2 노드가 수신된 메시지에 기초하여 자신의 부모 노드와 상기 인터넷 게이트웨이 사이의 제1 링크 수 및 상기 제1 노드와 상기 인터넷 게이트웨이 사이의 제2 링크 수를 비교하는 단계; 및

상기 비교 결과에 따라 상기 제2 노드가 상기 제1 노드를 상기 자신의 예비 부모 노드로 추가 또는 자신의 부모 노드로 설정하는 단계;를 포함하는 것을 특징으로 하는 트리 형성 방법.

청구항 2

제1항에 있어서,

상기 설정하는 단계는,

상기 비교 결과, 상기 제1 링크 수 및 상기 제2 링크 수가 동일하면 상기 제2 노드가 상기 제1 노드를 자신의 예비 부모 노드로 추가하는 것을 특징으로 하는 트리 형성 방법.

청구항 3

제1항에 있어서,

상기 설정하는 단계는,

상기 비교 결과, 상기 제1 링크 수가 상기 제2 링크 수보다 크면, 상기 제1 노드를 부모 노드로 설정하고 자신의 부모 노드와 연결을 해제하는 것을 특징으로 하는 트리 형성 방법.

청구항 4

제1항에 있어서,

상기 적어도 하나의 노드를 마련하는 단계는,

상기 복수의 노드 각각이 상기 인터넷 게이트웨이에 의해 브로드캐스트된 메시지를 수신하고, 수신된 메시지에 응답하여 링크 연결 요청 메시지를 인터넷 게이트웨이에 전송하는 단계; 및

상기 링크 연결 요청 메시지에 대한 응답으로 확인 메시지를 수신하면, 상기 복수의 노드 각각이 상기 인터넷 게이트웨이에 연결되는 단계;를 포함하는 것을 특징으로 하는 트리 형성 방법.

청구항 5

제4항에 있어서,

제3 노드가, 상기 복수의 노드 중 하나가 인터넷 게이트웨이에 유니캐스트한 메시지를 수신하고 상기 수신된 메시지에 응답하여 링크 연결 요청 메시지를 상기 복수의 노드 중 하나에 전송하는 단계; 및

상기 링크 연결 요청 메시지에 대한 응답으로 확인 메시지가 수신되면, 상기 제3 노드가 상기 복수의 노드 중 하나에 연결되는 단계;를 더 포함하는 것을 특징으로 하는 트리 형성 방법.

청구항 6

제1항에 있어서,

상기 제2 노드가 자신의 부모 노드와 연결이 해제되고, 자신의 자식 노드로부터 메시지를 수신하는 단계;

상기 메시지를 수신한 상기 제2 노드가 자신의 부모 노드로 연결 해제 메시지를 전송하는 단계; 및

상기 연결 해제 메시지에 대응하여, 상기 제2 노드의 자식 노드가 새로운 노드를 부모 노드로 설정하는 단계;를 더 포함하는 것을 특징으로 하는 트리 형성 방법.

청구항 7

제1항에 있어서,

상기 제2 노드에 연결된 고아 노드가 상기 고아 노드와 상기 인터넷 게이트웨이 사이의 링크 수를 변경하는 단계;

상기 고아 노드가 자식 노드를 갖는 경우 상기 제2 노드에 메시지를 전송하는 단계;

상기 메시지를 수신한 자식 노드가 상기 자식 노드와 상기 인터넷 게이트웨이 사이의 링크 수를 변경하는 단계; 및

상기 변경된 링크 수가 기 설정된 값을 초과하면, 상기 자식 노드가 상기 고아 노드와의 연결을 해제하는 단계;를 포함하는 것을 특징으로 하는 트리 형성 방법.

명 세 서

기 술 분 야

[0001] 본 발명은 트리 형성 방법에 관한 것으로, 보다 상세하게는 이동성 애드 혹 네트워크에서의 트리 형성 방법에 관한 것이다.

배 경 기 술

[0002] 이동 애드 혹 네트워크(Mobile ad hoc networks : MANETs)는 관리하고 사용하기 간단하기 때문에 많은 영역에 적용할 수 있다. 하지만, 이동 노드(Mobile nodes : MNs)가 인터넷 액세스 혹은 다른 MANET의 MN과 통신을 필요로 하는 경우에는 적용되기 어렵다. 비록 무선 1-홉을 통해 Mobile IP가 다른 네트워크에 성공적으로 적용되어도([1] Perkins, C. (1996). “IP mobility support,” request for comments (Standard) 2002, Internet Engineering Task Force, Oct 1996.), MN들을 무선 멀티-홉을 통해 인터넷 게이트웨이(Internet Gateway :IG) 또는 기지국(base station)에 등록하고 등록된 MN을 유지하는 것(이동성 관리(mobility management))은 과도한 제어 오버헤드(control overhead)를 포함하는 문제를 겪게 된다.

[0003] 이에 따라, 이동성 관리가 요구되는 MANET의 인터넷 연결 규정에 대한 많은 연구가 실행되었다. 먼저, Broch는 IG가 2개의 네트워크 인터페이스 카드를 가지는 것으로 가정했다. 하나는 표준 IP 메커니즘을 실행하여 인터넷에 대한 브릿지(bridge)로서 동작하고, 다른 하나는 MNs에 대한 브릿지로서 동작한다([2] Broch, J., Maltz, D. A., & Johnson, D. B.(1999). Supporting hierarchy and heterogeneous interfaces in multi-hop wireless adhoc networks. In Proc. Int’l. Symp. Parallel Architecture, Algorithms, and Networks, Perth, Australia (pp. 370-375), June 1999.). Jonsson은 제어 오버헤드를 줄이기 위해 MIPMANET 셀 스위칭 알고리즘(MMIPMANET cell switching algorithm)을 사용하는 IG로 최적화된 MN의 등록 타이밍에 초점을 맞췄다([3] Jonsson, D., Alriksson, F., Larsson, T., Johansson, P., & Maguire, J. G. (2000). MIPMANET-mobile IP for mobile ad hoc networks. In Proceedings of IEEE/ACM workshop on mobile and adhoc networking and computing (MobiHoc’00), Boston, MA, USA, pp. 75-85, Aug 2000). Sun은 MNs와 IGs 사이의 멀티-홉 경로(path)를 찾기 위해 이동 IP와 AODV([4] Perkins, C. E., & Royer, E. M. (1999). Ad-hoc on-demand distance vector routing. In Second IEEE workshop on mobile computing systems and applications (pp. 90-100), Feb 1999.)를 통합하려고 했다([5] Sun, Y., Belding-Royer, E. M., & Perkins, C. E. (2002). Internet connectivity for ad hoc mobile networks. International Journal of Wireless Information Networks special issue on Mobile Ad Hoc Networks (MANETs): Standards, Research, and Applications, 9(2), 75-88.). 또한, Ratanchandani은 에이전트 광고기법(agent advertisement), TTL 스코핑, 에이전트 광고기법, 에이전트 요청(Agent Solicitation)의 검색과 같은 기술들을 결합하는 하이브리드 방식(hybrid scheme)을 제안했다([[6] Ratanchandani, P., Kravets, R. (2003). A hybrid approach to Internet connectivity for mobile ad

hoc networks. In Proceeding of IEEE WCNC 2003 (pp. 1522-1527), March 2003). Ammari는 이동 IP와 DSDV([7] Perkins, C. E., & Bhagwat, P. (1994). Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. ACM SIGCOMM: Computer Communications Review, 24(4), 234-244.)를 사용한 3 레이어 접근(three layer approach) 방법을 제안했다([8] Ammari, H., & El-Rewini, H. (2004). Integration of mobile ad hoc networks and the Internet using mobile gateways. In Proceedings of 18th international symposium of parallel and distributed processing, vol. 13 (pp. 218b), April 2004.). 이는 Mss와 IG 사이에서 인터페이스로서 모바일 게이트웨이를 사용한다. 한편, Moshrify 및 Ruiz는 AODV에 기반한 게이트웨이 발견 방식(gateway discovery shcemes)을 연구했다([9] El-Moshrify, H., Mangoud, M. A., Rizk, M. (2007). Gateway discovery in ad hoc ondemand distance vector (AODV) routing for Internet connectivity. In Radio science conference, NRSC 2007 (pp. 1-8), March 2007, [10] Ruiz, P., Gomez-Skarmeta, A. (2004). Enhanced internet connectivity for hybrid ad hoc networks through adaptive gateway discovery. In Proceedings of the 29th annual IEEE international conference on local computer networks (LCN' 04), IEEE computer society, Los Alamitos, CA, USA (pp. 370-377), Nov 2004.). 이러한 이동성 관리방식은 제한적이기는 하지만, 이동성 관리를 위한 광고기법(advertisement) 및/또는 요청(solicitation)과 같은 제어 메시지를 전송하기 위하여 플러딩(flooding) 기법을 사용하고 있다.

[0004] 이동성 관리에 대한 종래 접근들은 프로액티브 방식(Proactive shcemes)([9]), 리액티브 방식(reactive shcemes)([2, 6, 9]) 및 하이브리드 방식(hybrid schemes)([6], [9])의 3가지 카테고리로 분류될 수 있다.

[0005] 프로액티브 방식에서, IG는 광고 메시지를 네트워크에 주기적으로 보내 등록 경로를 형성한다. 이에 따라, MN은 규칙적인 인터벌에서 등록 경로를 따라 IG에 등록한다. 이러한 방법은 통신에 참가하는지와 무관하게 모든 MNs를 등록하기 때문에, 원격 호스트는 어느 때나 연결 요청을 생성할 수 있는 장점이 있다. 하지만, MN이 등록을 초기화하는 다음 인터벌이 시작하기 전에 등록 경로가 끊어질 수 있기 때문에, MN은 등록을 실패할 수 있는 문제가 있다.

[0006] 리액티브 방식에서, MN이 인터넷을 통해 원격 호스트와 통신을 수행하기를 원할 때, IG를 검색하기 위한 요청 메시지를 인터넷에 플러딩한다. MN 등록 경로가 설정되었다는 확인 메시지를 수신하면, 그 경로를 따라 등록을 수행한다.

[0007] 하이브리드 방식은 다른 2개의 접근들의 단점을 극복하기 위해 제안되었다. 이 접근에서, IG는 홉의 제한된 범위 내에서(k 홉) 광고 메시지를 네트워크로 플러딩한다. 여기서, k 홉 밖의 MNs은 IG에 대한 경로를 가지는 등록된 MN의 위치를 찾기 위해 요청 메시지를 네트워크에 보내며, 등록된 노드를 통해 IG에 등록할 수 있다. 이러한 접근은 플러딩 범위를 줄일 수 있으나, 제한적 플러딩을 사용하는 문제가 있다.

발명의 내용

해결하려는 과제

[0008] 본 발명은 상기와 같은 문제점을 해결하기 위하여 안출된 것으로서, 본 발명의 목적은, 이동성 애드 혹 네트워크를 형성하는 노드들이 복수의 부모 노드를 가지는 방법을 제공함에 있다.

과제의 해결 수단

[0009] 이상과 같은 목적을 달성하기 위한 본 발명의 실시 예에 따른 이동성 애드 혹 네트워크(Mobile ad hoc networks : MANETs)에서의 트리 형성 방법은, 인터넷 게이트웨이(internet gateway)에 연결된 복수의 노드 및 상기 복수의 노드 각각에 순차적으로 연결된 적어도 하나의 노드를 마련하는 단계, 상기 적어도 하나의 노드 중 제1 노드가 자신의 부모 노드로 유니캐스트한 메시지를, 상기 제1 노드의 인접 노드인 제2 노드가 수신하는 단계, 상기 제2 노드가 수신된 메시지에 기초하여 자신의 부모 노드와 상기 인터넷 게이트웨이 사이의 제1 링크 수 및 상기 제1 노드와 상기 인터넷 게이트웨이 사이의 제2 링크 수를 비교하는 단계 및 상기 비교 결과에 따라 상기 제2 노드가 상기 제1 노드를 상기 자신의 예비 부모 노드로 추가 또는 자신의 부모 노드로 설정하는 단계를 포함한다.

[0010] 또한, 상기 설정하는 단계는, 상기 비교 결과, 상기 제1 링크 수 및 상기 제2 링크 수가 동일하면 상기 제2 노드가 상기 제1 노드를 자신의 예비 부모 노드로 추가할 수 있다.

[0011] 또한, 상기 설정하는 단계는, 상기 비교 결과, 상기 제1 링크 수가 상기 제2 링크 수보다 크면, 상기 제1 노드

를 부모 노드로 설정하고 자신의 부모 노드와 연결을 해제할 수 있다.

[0012] 또한, 상기 적어도 하나의 노드를 마련하는 단계는, 상기 복수의 노드 각각이 상기 인터넷 게이트웨이에 의해 브로드캐스트된 메시지를 수신하고, 수신된 메시지에 응답하여 링크 연결 요청 메시지를 인터넷 게이트웨이에 전송하는 단계 및 상기 링크 연결 요청 메시지에 대한 응답으로 확인 메시지를 수신하면, 상기 복수의 노드 각각이 상기 인터넷 게이트웨이에 연결되는 단계를 포함한다.

[0013] 또한, 제3 노드가, 상기 복수의 노드 중 하나가 인터넷 게이트웨이에 유니캐스트한 메시지를 수신하고 상기 수신된 메시지에 응답하여 링크 연결 요청 메시지를 상기 복수의 노드 중 하나에 전송하는 단계 및, 상기 링크 연결 요청 메시지에 대한 응답으로 확인 메시지가 수신되면, 상기 제3 노드가 상기 복수의 노드 중 하나에 연결되는 단계를 더 포함할 수 있다.

[0014] 또한, 상기 제2 노드가 자신의 부모 노드와 연결이 해제되고, 자신의 자식 노드로부터 메시지를 수신하는 단계, 상기 메시지를 수신한 상기 제2 노드가 자신의 부모 노드로 연결 해제 메시지를 전송하는 단계 및 상기 연결 해제 메시지에 대응하여, 상기 제2 노드의 자식 노드가 새로운 노드를 부모 노드로 설정하는 단계를 더 포함할 수 있다.

[0015] 또한, 상기 제2 노드에 연결된 고아 노드가 상기 고아 노드와 상기 인터넷 게이트웨이 사이의 링크 수를 변경하는 단계, 상기 고아 노드가 자식 노드를 갖는 경우 상기 제2 노드에 메시지를 전송하는 단계, 상기 메시지를 수신한 자식 노드가 상기 자식 노드와 상기 인터넷 게이트웨이 사이의 링크 수를 변경하는 단계 및 상기 변경된 링크 수가 기 설정된 값을 초과하면, 상기 자식 노드가 상기 고아 노드와의 연결을 해제하는 단계를 포함한다.

발명의 효과

[0016] 본 발명의 다양한 실시 예에 따르면, 이동성 애드 혹 네트워크를 형성하는 노드들 각각은 복수의 부모 노드를 가질 수 있다. 이에 따라, 노드가 자신의 부모 노드와 연결이 끊어져도 별도의 제어 오버헤드 없이 인터넷 게이트웨이에 대한 경로를 유지할 수 있게 된다.

도면의 간단한 설명

[0017] 도 1은 본 발명의 일 실시 예에 따른, 쿼지-트리 네트워크를 설명하기 위한 도면,

도 2 내지 6은 본 발명의 일 실시 예에 따른, 쿼지-트리 네트워크를 형성하는 노드들의 연결방법을 설명하기 위한 도면,

도 3은 본 발명의 일 실시 예에 따른, 인터넷 게이트웨이와 노드간의 링크 형성 방법을 설명하기 위한 도면이다.

도 4는 본 발명의 일 실시 예에 따른, 노드들 간의 링크 형성 방법을 설명하기 위한 도면이다.

도 5는 본 발명의 일 실시 예에 따른, 노드들이 다중 부모 노드와 링크를 형성하는 방법을 설명하기 위한 도면이다.

도 7은 본 발명의 일 실시 예에 따른, 트리 노드가 부모 노드를 변경하는 방법을 설명하기 위한 도면이다.

도 8은 본 발명의 일 실시 예에 따른 트리 노드가 자식 노드와의 링크를 해제하는 방법을 설명하기 위한 도면이다.

도 9는 트리 노드에 있어 루프(loop)가 형성되는 경우를 설명하기 위한 도면이다.

도 10은 본 발명의 일 실시 예에 따른 루프를 탐색 및 제거하는 방법에 대한 알고리즘을 나타낸 도면이다.

도 11a 내지 도 11c는 본 발명의 일 실시 예에 따른 기본 전개 시나리오를 설명하기 위한 도면들이다.

도 12는 TB 및 QT의 nSpeed를 비교하기 위한 도면이다.

도 13 내지 16은 다양한 시나리오에 따라 다양한 이동성 관리 프로토콜을 비교하기 위한 도면이다.

도 17 내지 20은 nNode가 가변함에 따른 프로토콜을 비교하는 도면들이다.

도 21 내지 24는 mSpeed의 가변에 따른 프로토콜을 비교하는 도면들이다.

도 25 내지 28은 본 발명의 실시 예에 따른 시나리오에서 QT 및 QT-LDR을 비교하는 도면들이다.

도 29는 본 발명의 일 실시 예에 따른 트리 구조 형성 방법을 설명하기 위한 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0018] 이하에서는 도면을 참조하여 본 발명을 보다 상세하게 설명한다.

[0019] 도 1은 본 발명의 이해를 돕기 위한 트리 구조를 설명하기 위한 도면이다.

[0020] 도 1을 참조하면, 트리 구조는 인터넷과 같은 네트워크에 연결된 인터넷 게이트웨이(internet gateway :IG) 및 복수의 노드들(1 내지 22)을 포함하며, 각 트리는 인터넷 게이트웨이에서 시작한다.

[0021] 설명의 편의를 위해, 노드들 사이의 관계를 정의하면 다음과 같다.

[0022] 먼저, 노드들 사이에 링크(link)를 형성할 때, 노드가 제일 먼저 연결하는 상위 노드를 부모 노드(parent node)라 하고, 부모 노드에 연결된 하위 노드를 자식 노드(child node)라고 하며, 부모 노드와 자식 노드가 연결된 링크를 트리-링크(tree-link)라 한다. 한편, 노드가 자식 노드를 가지고 있어도 부모 노드를 가지고 있지 않다면, 그 노드를 고아 노드(orphan node)라 한다.

[0023] 또한, 인터넷 게이트웨이를 기점으로, 상위 노드에서 하위 노드로 순차적으로 형성된 구조를 트리(tree)라 하며, 트리에 포함된 노드를 트리 노드(tree node)라 언급한다. 그리고, 인터넷 게이트웨이의 자식 노드 중 하나에서 시작하는 트리를 IG-서브트리(IG-subtree)로 정의한다.

[0024] 도 1을 참조하면, 실선은 트리-링크를 나타내며, 점선은 트리에 속하지 않은 링크를 나타낸다. 인터넷 게이트웨이 1과 인터넷 게이트웨이 2로부터 생성된 2개의 트리 및 1, 7, 12, 19로부터 생성된 4개의 IG-서브트리가 존재함을 알 수 있다.

[0025] 도 2는 본 발명의 일 실시 예에 따른, 쿼지-트리 네트워크를 설명하기 위한 도면이다. 여기서, 쿼지-트리(quasi-tree)란, 이동성 애드 혹 네트워크를 구성하는 노드들 각각이 복수의 부모 노드를 가지는 트리를 의미한다.

[0026] 도 2를 참조하면, 쿼지-트리 네트워크는 하나의 인터넷 게이트웨이(internet gateway :IG)(14) 및 복수의 트리 노드들(1 내지 13)을 포함할 수 있다. 여기서, 복수의 트리 노드들 각각은 무선 인터페이스를 사용하여 서로 통신하며, 이에 따라, 다른 노드 사이에서 전송되는 패킷을 수신할 수 있다.

[0027] 또한, 복수의 트리 노드들 각각은 인터넷과 같은 네트워크와 연동하기 위해, 부모 노드를 통해 인터넷 게이트웨이에 등록할 수 있다.

[0028] 또한, 복수의 트리 노드들 각각은 이동성 노드(mobile node :MN)이기 때문에, 이동이 자유롭다.

[0029] 도 2에서, 복수의 트리 노드들 각각은 복수의 부모 노드를 가지는 것을 알 수 있다. 구체적으로, 트리 노드 각각은 하나의 주 부모 노드(primary parent node)와 0 또는 그 이상의 예비 부모 노드(ordinary parent node)를 포함하는 다중 부모 노드를 가질 수 있다.

[0030] 복수의 트리 노드들 각각이 고아 노드일 때, 트리에 첫 번째로 링크하는 상위 노드를 주 부모 노드라고 하며, 인터넷 게이트웨이까지의 링크의 수가 주 부모 노드와 동일한 노드를 예비 부모 노드라고 할 수 있다.

[0031] 이처럼, 쿼지 트리 네트워크에서는, 트리 노드는 다중 부모 노드를 가지기 때문에 비록 노드가 주 부모 노드와의 링크가 끊어져도, 핸드쉐이킹 프로세스(handshaking process) 비용 없이 예비 부모 노드를 통해 인터넷 게이트웨이까지의 업스트림 경로(upstream path)를 유지할 수 있게 된다.

[0032] 이에 따라, 트리 노드들 각각은 모든 부모 노드와의 링크가 끊어진 경우에만 다른 트리 노드에 연결을 시도하기 때문에, 제어 오버헤드(comtrol overhead)를 감소시킬 수 있다.

[0033] 또한, 쿼지-트리 네트워크에서는 제어 오버헤드를 감소시키기 위해 딜레이드-조인 기법(delayed-join principle)을 사용하며, 이에 따라, 고아 노드는 이웃한 트리 노드로부터 헬로우 메시지를 수신할 때까지 기다려야 한다. 즉, 이웃한 트리 노드들이 그들의 부모 노드들에게 보낸 헬로우 메시지를 수신하면, 고아 노드는 해당 노드에 링크를 형성하려고 한다. 이러한 방식은 제어 오버헤드를 감소시키고, 메시지 충돌 확률을 증가시키는 브로드캐스팅의 사용을 제거할 수 있다. 또한, 유니캐스팅은 IEEE 802.11 MAC 프로토콜에서 이용되는 ACK 메시지를 이용하여 신속히 링크 무효성을 결정할 수 있다.

[0034] 또한, 쿼지-트리 네트워크에서, 트리 노드들 각각은 가장 짧은 링크 수를 가지는 다중 부모 노드들을 통해 인터

넷 게이트웨이에 연결되기 때문에, 통신 오버헤드 및 부모 스위칭 오버헤드를 줄일 수 있다.

[0035] 이하에서, 이러한 쿼지-트리를 형성하는 방법에 대해서 첨부된 도면을 참조하여 보다 상세히 설명한다.

[0036] 이를 위해, 각 노드들이 전송하는 제어 메시지(control message : CT)를 아래와 같이 정의할 수 있다.

[0037] - 헬로우(hello)=(CT, HopToIG) : 인터넷 게이트웨이는 그의 자식 노드에 헬로우 메시지를 주기적으로 유니캐스트(unicast)하며, 단지, 자식 노드를 가지지 않을 때 헬로우 메시지를 브로드캐스트(broadcast)한다. 다만, 노드는 그의 주 부모 노드에게만 헬로우 메시지를 유니캐스트한다.

[0038] - J-REQ=(CT) : 노드가 이웃한 노드에 연결하기 위해 J-REQ 메시지를 전송한다.

[0039] - CR-REQ=(CT) : 고아 노드가 헬로우 메시지를 보낸 자식 노드에 이러한 CR-REQ 메시지로 응답한다.

[0040] - REG=(CT, node ID) : 노드가 인터넷 게이트웨이에 등록하기 위해 이러한 REG 메시지를 전송한다. REG 메시지에는 트리-기반 접근 방식에서 트리 유지 정보를 포함한다.

[0041] - SOL=(CT, node ID) : 리액티브 또는 하이브리드 방식을 사용할 때, 인터넷 게이트웨이에 등록되길 원하는 노드는 네크워크를 통해 이러한 SOL 메시지를 전송한다.

[0042] - ADV=(CT, node ID) : 프로액티브 또는 하이브리드 방식을 사용할 때, 인터넷 게이트웨이는 이러한 ADV 메시지를 주기적으로 전송한다.

[0043] 아래의 표 1은 제어 메시지의 정의를 요약한 것이다.

표 1

[0044]

Messages	Usage
Hello = (CT, HopToIG)	IG unicasts this hello message to one of its children periodically; it broadcasts this message only when it does not have any children (a very rare situation). However, the MN periodically unicasts this message only to its *primary parent*
J-REQ = (CT)	MN sends this *join request* message to its neighbor to join
CR-REQ = (CT)	An orphan-node responds with this *children-release request* message to the child that has sent a Hello message
REG = (CT, NodeID)	A node sends this *registration* message to register with an IG. Registration includes tree maintenance in a tree-based approach
SOL = (CT, NodeID)	A node that wants to register with the IG floods the network with this *solicitation* message when a reactive or hybrid scheme is used
ADV = (CT, NodeID)	IG sends this *advertisement* message periodically when a proactive or a hybrid scheme is used

[0045] 그리고, 타이머(timer)는 아래와 같이 정의될 수 있다.

[0046] - 헬로우-타이머 : 헬로우 메시지를 주기적으로 전송하기 위해 사용된다.

[0047] -p-c 타이머 : 부모 노드와 자식 노드 사이의 유효한 기간을 나타내기 위해 사용된다. 이러한 p-c 타이머가 경과하면, 부모 노드와 자식 노드를 연결하는 링크는 끊어진다.

[0048] -경로 타이머(path timer) : 인터넷 게이트웨이로부터 노드에 형성된 경로의 유효 기간을 나타내기 위해 사용된다.

[0049] -등록-타이머(registration timer) : 등록된 노드의 유효한 기간을 나타내기 위해 사용된다. 인터넷 게이트웨이

가 노드로부터 REG 메시지를 수신하면, 인터넷 게이트웨이는 해당 노드에 대한 등록-타이머를 리셋(reset)한다. 또한, 노드가 인터넷 게이트웨이에 REG 메시지를 성공적으로 전송하면, 해당 노드에 대한 등록-타이머는 리셋(reset)된다. 만일 등록-타이머가 경과되면, 인터넷 게이트웨이는 해당 노드의 등록을 제거한다.

[0050] -광고-타이머(advertisement-timer) : ADV 메시지를 주기적으로 전송하기 위해 사용된다.

[0051] 아래의 표 2는 타이머의 정의를 요약한 것이다.

표 2

[0052]

Timers	Usage
Hello-timer	Used to trigger the hello periodically
p–c-timer	Used to indicate the valid duration of parent–child relationship. If this timer expires, the corresponding link is removed from the parent and the child
Path-timer	Used to indicate the valid duration of an established path from IG to MN or vice versus
Registration-timer	Used to indicate the valid duration of a registered node. Whenever an IG receives REG from an MN, the IG resets this timer for the MN. Whenever an MN sends REG to an IG successfully, it resets this timer. If the timer expires, the IG removes the MN
Advertisement-timer	Used to trigger the ADV message periodically

[0053] 이하에서는 쿼지-트리 정보 구조에 대해 간략히 설명하도록 한다.

[0054] 아래의 수학식 1에서 i.IG가 노드 i가 속하는 IG이고, i.NS, i.P, i.C는 각각 노드 i의 이웃, 부모, 자식 노드 집합을 나타낸다.

[0055] 쿼지-트리 정보 구조(QTIS : Quasi-Tree Information Structure)는 아래의 수학식 1과 같이 정의된다.

수학식 1

[0056]
$$\begin{aligned} \mathrm{QTIS}^{(i)} &= (i.P,\ i.C, i.HopToIG) \text{ where } i.P \\ &= \{x | x \in i.NS, x.HopToIG = i.HopToIG - 1\}, i.C \\ &= \{x | x \in i.NS, x.HopToIG = i.HopToIG + 1\}. \end{aligned}$$

[0057] 여기서, i.HopToIG는 노드 i의 HopToIG를 나타낸다. 모든 노드는 이러한 QTIS를 유지하고, 어떠한 자손도 갖지 않는 고아 노드에 대한 QTIS=(Ø Ø ∞)이다.

[0058] 도 3은 본 발명의 일 실시 예에 따른, 인터넷 게이트웨이와 노드간의 링크 형성 방법을 설명하기 위한 도면이다.

[0059] 도 3에서, 트리에 링크되기 전인, 고아 노드(320)는 인터넷 게이트웨이(310)가 브로드캐스트하거나, 그의 자식 노드에 유니캐스트한 헬로우 메시지를 수신한다. 이에 따라, 고아 노드(320)는 인터넷 게이트웨이(310)에 J-REQ 메시지를 전송하고, 이에 대응하여 확인 메시지를 인터넷 게이트웨이(310)로부터 수신하면 고아 노드(320)는 인터넷 게이트웨이(310)에 연결되어 트리 노드를 구성하며, 인터넷 게이트웨이(310)의 자식 노드가 되게 된다.

[0060] 도 4는 본 발명의 일 실시 예에 따른, 노드들 간의 링크 형성 방법을 설명하기 위한 도면이다.

[0061] 도 4에서, 고아 노드(430)가 트리 노드(420)에 연결되는 경우를 상정한다. 트리 노드(420)가 인터넷 게이트웨이

(410)에 유니캐스트하는 헬로우 메시지를 고아 노드(430)가 수신하면, 고아 노드(430)는 트리 노드(420)에 J-REQ 메시지를 전송한다. 이에 대한 응답으로, 트리 노드(420)로부터 확인 메시지를 수신하면, 고아 노드(430)는 트리 노드(420)에 연결되어 트리를 형성하게 되며, 트리 노드(420)는 고아 노드(430)를 그의 자식 노드로 설정하게 된다.

[0062] 도 5는 본 발명의 일 실시 예에 따른, 노드들이 다중 부모 노드와 링크를 형성하는 방법을 설명하기 위한 도면이다.

[0063] 도 5를 참조하면, 모든 노드들은 트리 노드에 해당한다. 또한, 트리 노드(520)는 인터넷 게이트웨이(510)와 직접 링크되어 있으며, 트리 노드(530) 및 트리 노드(540)의 주 부모 노드는 트리 노드(520)이며, 트리 노드(550)의 주 부모 노드는 트리 노드(530)이다.

[0064] 도 5에서, 트리 노드(540)가 자신의 주 부모 노드인 트리 노드(520)에 전송한 헬로우 메시지를 트리 노드(550)가 수신하면, 트리 노드(550)는 자신의 주 부모 노드의 어드레스와 트리 노드(540)의 어드레스를 비교한다. 구체적으로, 트리 노드(550)는 트리 노드(530)가 인터넷 게이트웨이에 연결되기 위한 링크 수와 트리 노드(540)가 인터넷 게이트웨이에 연결되기 위한 링크 수를 서로 비교한다. 즉, 인터넷 게이트웨이까지의 홉의 수(HopToIG)를 비교하게 된다.

[0065] 따라서, 헬로우 메시지에는 트리 노드가 인터넷 게이트웨이에 연결되기 위한 링크 수에 대한 정보가 포함될 수 있으며, 트리 노드 각각은 자신의 주 부모 노드가 인터넷 게이트웨이에 연결되기 위한 링크 수에 대한 정보를 미리 알고 있음이 바람직하다.

[0066] 그 결과, 트리 노드(530)와 트리 노드(540) 각각이 가지는 인터넷 게이트웨이에 연결되기 위한 링크 수가 동일하다면, 트리 노드(550)는 트리 노드(540)를 자신의 예비 부모 노드로 설정하며, 트리 노드(540)는 트리 노드(550)를 자신의 자식 노드로 설정할 수 있다. 한편, 이러한 동작들은 서로 독립적으로 이루어질 수 있다.

[0067] 한편, 본 발명의 일 실시 예에 따른 쿼지 트리 네트워크에서, 트리 노드들 각각은 다중 부모 노드들을 통해 가장 짧은 링크 수를 가지며 인터넷 게이트웨이에 연결됨은 전술한바 있다. 이는 다중 부모 노드와 자식 노드 사이의 p-c 타이머에 의해 제어된다.

[0068] 구체적으로, 트리 노드들 각각은 그들의 자식 노드로부터 헬로우 메시지를 수신할 때마다, 해당 자식 노드와 관련된 p-c 타이머를 리셋한다. 만일 소정의 트리 노드가 HopToIG를 변경한 자식 노드 중 하나로부터 헬로우 메시지를 수신하면, 해당 자식 노드와의 연결을 해제한다.

[0069] 또한, 소정의 트리 노드가 그들의 부모 노드로부터 헬로우 메시지를 수신하면, 해당 부모 노드와 관련된 타이머를 리셋한다. 만일, HopToIG가 증가된 주 부모 노드로부터 헬로우 메시지를 수신하면, 해당 트리 노드는 예비 부모 노드들 중 하나를 주 부모 노드로 설정한다. 만일, 예비 부모 노드를 가지고 있지 않다면, 주 부모 노드의 HopToIG를 수신된 새로운 HopToIG에 따라 업데이트한다. 만일, 트리 노드가 HopToIG가 감소된 예비 부모 노드로부터 헬로우 메시지를 수신하면, 해당 예비 부모 노드를 주 부모 노드로 설정하고 기존의 주 부모 노드와의 링크를 해제할 수 있다.

[0070] 이러한 방법에 의해, 트리 노드 각각은 인터넷 게이트웨이까지 최단 링크 수를 유지할 수 있게 되는 것이며, 도 6은 본 발명의 일 실시 예에 따른 트리 노드 각각과 부모 노드와의 관계를 설정하는 자세한 알고리즘을 도시한다.

[0071] 또한, 본 발명의 일 실시 예에 따른, 트리 노드 각각은 링크의 무효성을 빨리 판단할 수 있다. 즉, 트리 노드가 기 설정된 구간 동안 다중 부모 노드들을 제외한 다른 노드들로부터 헬로우 메시지를 수신하지 못하면, 해당 트리 노드는 그들의 다중 부모 노드와 링크를 유지한다. 여기서, 기 설정된 기간은 트리 노드와 그들의 다중 부모 노드들 사이의 헬로우 메시지 송신 기간을 의미할 수 있다.

[0072] 도 7은 본 발명의 일 실시 예에 따른, 트리 노드가 부모 노드를 변경하는 방법을 설명하기 위한 도면이다.

[0073] 도 7을 참조하면, 인터넷 게이트웨이(710)에 직접 링크된 트리 노드(720) 및 트리 노드(750)가 존재하며, 트리 노드(730) 및 트리 노드(740)는 주 부모 노드로서 트리 노드(720) 및 트리 노드(730)를 각각 가지고 있다.

[0074] 도 7에서, 트리 노드(750)가 인터넷 게이트웨이(710)로 유니캐스트하는 헬로우 메시지를 트리 노드(740)를 수신하면, 트리 노드(740)는 트리 노드(730)와 트리 노드(750)의 인터넷 게이트웨이(710)까지 가지는 링크 수를 비교한다. 트리 노드(750)가 가지는 링크 수가 트리 노드(730)보다 짧기 때문에, 트리 노드(740)는 트리 노드

(730)와의 링크를 해제하고, 트리 노드(750)를 주 부모 노드로 설정할 수 있다. 이 후, 트리 노드(720)가 인터넷 게이트웨이(710)로 유니캐스트하는 헬로우 메시지를 트리 노드(740)가 수신하면, 트리 노드(740)는 트리 노드(720)를 그의 예비 부모 노드로 설정할 수 있다.

[0075] 한편, 트리 노드(750)가 인터넷 게이트웨이(710)로 유니캐스트하는 헬로우 메시지를 트리 노드(730)가 수신하였다면, 트리 노드(730)는 트리 노드(750)를 그의 예비 부모 노드로 설정할 수 있다.

[0076] 이러한 방법에 의해, 트리 노드 각각은 인터넷 게이트웨이까지 최단 링크 수를 유지할 수 있게 된다.

[0077] 도 8은 본 발명의 일 실시 예에 따른 트리 노드가 자식 노드와의 링크를 해제하는 방법을 설명하기 위한 도면이다.

[0078] 도 8을 참조하면, 트리 구조가 제시되며, 예비 부모 노드를 가지지 못하는 노드(820)는 주 부모 노드(810)와 링크가 해제되어 고아 노드가 된 것을 알 수 있다. 이는 노드(820)가 주 부모 노드(810)에 헬로우 메시지를 전송하는데 실패한 경우에 해당될 수 있다.

[0079] 이 경우, 고아 노드(820)는 그의 자식 노드인 트리 노드(840)에게 바로 CR-REQ 메시지 전송하지 않는다. 이는 네트워크에서 불필요한 오버헤드를 야기할 수 있기 때문이다. 고아 노드(820)는 트리 노드(840)로부터 헬로우 메시지를 수신하면, 그때 CR-REQ 메시지를 전송한다. 이에 대응하여, 트리 노드(840)는 예비 부모 노드인 트리 노드(830)를 주 부모 노드로 설정하며, 고아 노드(820)와의 연결을 해제할 수 있다. 한편, 트리 노드(840)가 예비 부모 노드를 가지고 있지 않았다면, 노드(820)와 마찬가지로, 고아 노드가 된다.

[0080] 한편, 노드들 사이에서 루프는 토폴로지 변경의 업데이트에 있어 동기화 지연(synchronization delay) 때문에, 고아 노드가 트리 노드에 잘못 연결되어 형성될 수 있다. 여기서, 동기화 지연이란, 만일 서로 다른 두 개의 노드 사이에 링크가 형성되거나 해제될 때, 다른 노드가 일정한 기간이 경과된 후에 이를 알게 되어 발생하는 지연을 의미한다.

[0081] 이러한, 루프가 형성되면 루프에 포함된 노드가 루프가 파괴될 때까지 반복적으로 메시지를 전송하도록 하기 때문에, 네트워크 오버헤드를 야기한다.

[0082] 도 9는 트리 노드에 있어 루프(loop)가 형성되는 경우를 설명하기 위한 도면이다.

[0083] 도 9를 참조하면, 3개의 트리 노드(910, 920, 930)가 마련되며, 트리 노드(910)가 부모 노드와 링크가 해제된 것을 알기 전에, 트리 노드(930)가 트리 노드(920)에 연결되었다. 이 후, 부모 노드와 링크가 해제된 고아 노드(910)가 트리 노드(930)와 링크를 형성하여 노드들(910, 920, 930) 사이에 루프가 형성되게 되었다.

[0084] 본 발명의 일 실시 예에 따르면, 트리 깊이(tree depth)를 활용하여 생성된 루프를 탐지하고 분해한다. 이를 위해, 본 발명에서는 아래의 2가지 조건을 만족하면, 트리 노드가 그들의 부모 노드에게 HopToIG를 포함하는 헬로우 메시지를 즉시 전송하여 루프를 탐지할 수 있다.

[0085] - 트리 노드가 리프 노드(leaf node)가 아닌 경우

[0086] - 트리 노드 자신의 HopToIG가 변경된 경우

[0087] 먼저, 고아 노드가 트리 노드에 링크되는 경우를 가정하면, 고아 노드는 트리 노드가 될 것이고, 이 트리 노드의 HopToIG는 변경될 것이다. 이 경우, 고아 노드(트리 노드)가 자식 노드를 가지고 있다면(즉, 리프 노드가 아니라면), 고아 노드는 연결된 트리 노드(부모 노드)에 헬로우 메시지를 전송한다. 이에 따라, 고아 노드의 자식 노드가 헬로우 메시지를 수신하면, 그의 HopToIG를 변경한다. 이후, 자식 노드들 각각은 순차적으로 헬로우 메시지를 그들의 부모 노드에게 전송하고, 그의 HopToIG를 변경한다.

[0088] 이러한 과정은 상술한 2가지 조건이 더 이상 충족되지 않을 때까지 계속되며, 노드가 특정 깊이 값(DEPTH_LIMIT) 이상의 HopToIG를 가지는 것을 발견하면 노드는 루프에 포함되었다고 판단한다. 이 경우, 해당 노드가 다른 노드들과 링크를 해제함으로써 루프를 파괴할 수 있다.

[0089] 도 10은 본 발명의 일 실시 예에 따른 루프를 탐색 및 제거하는 방법에 대한 알고리즘을 나타낸 도면이다.

[0090] 도 10에 따른 알고리즘을 사용하면, 제한된 트리 사이즈에서 루프를 효과적으로 제한할 수 있다.

[0091] 본 알고리즘은 큰 깊이를 갖지만, 루프를 갖지 않는 트리가 겉으로 보기에 루프를 갖는 것으로 결정된 수 있기 때문에 매우 공격적으로 보일 수 있다. 하지만, 알고리즘은 트리의 사이즈를 제한하고 가능한 루프를 분해

(resolution)할 수 있고, 그럼으로써 트리 유지 오버헤드를 줄일 수 있다.

[0092] 한편, 본 발명에서는 QUALNet 3.9 시뮬레이터를 이용하여 두 가지 제안된 접근법, 즉, 쿼지-트리(QT) 방법 및 LDG를 포함하는 쿼지 트리 방법(QT-LDR)을 평가한다. 이러한 2개의 이동성 관리 방법은 프로액티브 방법(proactive), 리액티브 방법(reactive), 하이브리드 방법(hybrid), 트리 기반 방법(TB)를 포함하는 종래 의 이동성 관리 방법과 비교된다.

[0093] 이하에서는, 비교를 위해 종래 방법을 구현하는 방식을 설명한다. 모든 이동성 관리 프로토콜에서, 다른 리모트 MANETs에서 임의의 유선 리모트 호스트 또는 MN은 언제라도 MN에 연결 요청을 생성할 수 있다고 가정한다. 이에 따라, 비록 IG를 통해 다른 리모트 노드와 연결하기를 원하지 않아도, 모든 MN은 IG에 연결하기 위한 최선을 다해야 한다.

[0094] 모든 등록된 MN 및 그와 관련된 IG는 노드의 실효된(stale) 등록을 방지하기 위한 시간 동기된(time synchronized) 등록-타이머를 관리한다. 이에 따라, IG는 해당 등록-타이머가 종료되면, 노드 등록 리스트로부터 등록된 노드를 제거한다. 게다가, 모드 MN은 등록-타이머가 종료되면 IG에 주기적으로 등록해야 한다.

[0095] 등록-타이머는 모든 프로토콜에서 5-s로 설정된다. 다른 파라미터들은 각 프로토콜의 동작을 특성화하는데 이용된다. 키 파라미터에 이용되는 값들은 최고의 출력을 생성하기 위해 수번의 시뮬레이션을 통해 선택되었다.

[0096] **프로액티브 방법(Proactive method)**

[0097] IG는 광고(ADV) 메시지를 네트워크에 광고 타이머에 의해 특정 최대 TLL(본 시뮬레이션에서는 7을 사용함)을 사용하여 주기적으로 플러딩한다. 이러한 프로세스 동안, 모든 노드들은 IG에 리버스 경로(reverse path)를 형성하고, 리버스 경로를 통해 REG 메시지를 전송함으로써 IG에 등록된다. 모든 MN은 경로-타이머를 유지하며, 경로-타이머의 종료는 리버스 경로를 무효화시킨다. 만일 IG에 유효한 경로가 부족하여 만일 등록-타이머가 종료되면, 노드는 ADV 메시지를 기다린다. 본 시뮬레이션에서는 2-s의 광고 타이머와 3-s의 경로-타이머를 사용한다.

[0098] **리액티브 방법(Reactive method)**

[0099] 모든 노드들은 모든 써치가 실패할 때마다 TTL을 증가시키는 확장 링 써치 방법을 사용하여 IG를 써치하는 SOL(solicitation) 메시지를 네트워크에 플러딩한다([6] Ratanchandani, P., Kravets, R. (2003). A hybrid approach to Internet connectivity for mobile ad hoc networks. In Proceeding of IEEE WCNC 2003 (pp. 1522-1527), March 2003). 본 방법에서는, TTL은 1에서 시작하여 각각 써치 실패 동안 2 씩 증가한다. 써치는 IG를 찾거나 TTL이 특정 최대값(본 시뮬레이션에서는 7)에 도달할 때까지 증가한다. SOL 메시기 수신에 대하여, IG는 MN에 포워드 경로를 형성하고, 포워드 경로를 통해 MN에 SOL 확인 메시지를 전송하여 MN으로부터 IG에 리버스 경로를 형성한다. 만일 MN이 SOL 확인 메시지를 수신하면, MN은 리버스 경로를 통해 REG 메시지를 전송함으로써 IG에 등록된다. 노드는 IG에 대한 경로를 위해 경로-타이머를 관리한다. 노드는 만일 경로-타이머가 종료되면, SOL 메시지를 네트워크에 플러딩하여 IG에 등록된다. 본 시뮬레이션에서는, 3-s 경로-타이머를 사용한다.

[0100] **하이브리드 방법(Hybrid method)**

[0101] IG는 광고 타이머와 연관되어 k 홉에 ADV 메시지를 네트워크에 플러딩한다. 이러한 프로세스 동안, 수신하는 노드로부터 IG에 리버스 경로가 형성된다. 이러한 메시지를 수신하는 동안, 등록되지 않은 노드는 IG에 등록되기 위해 리버스 경로를 통해 REG 메시지를 전송한다. IG로부터 노드의 포워드 경로는 이러한 등록 프로세스를 통해 형성된다.형성된 경로 각각은 경로-타이머와 연관되며, 경로-타이머의 종료는 경로를 무효화시킨다.

[0102] k 홉 떨어져 ADV 메시지를 수신하지 못한 노드는 확장 링 써치 방법을 사용하여 등록된 노드 또는 IG를 검색하기 위한 SOL 메시지를 네트워크로 플러딩한다([6]). 이러한 프로세스에서, 수신한 등록된 노드로부터 SOL 메시지를 시작한 노들의 다른 포워드 경로가 형성된다. SOL 메시지를 수신하여, 등록된 노드는 SOL 메시지를 시작한 노드(시작 노드)에 응답 메시지를 전송하여, 등록되지 않은 시작 노드로부터 등록된 노드에 다른 리버스 경로를 형성한다. 결국, SOL 메시지를 전송한 노드는 2개의 리버스 경로가 결합된 경로를 따라 IG에 등록된다. 본 시뮬레이션에서, 2-s의 광고 타이머 및 3-s의 경로-타이머 및 k는 2로 설정한다.

[0103] **트리-기반 방법(Tree-based method)**

[0104] 트리 기반 방법은 헬로우 타이머를 사용하여 헬로우 메시지를 주기적으로 브로드캐스트하는 IG를 기반으로 한다([12] Han, T. D., Oh, H. (2009). A topology management routing protocol for mobile IP support of mobile ad hoc networks. In 8th international conference on ad hoc networks and wireless, Ad Hoc Now 2009, Murcia, Spain, LNCS 5793 (pp. 341-346), Sep 2009). 고아 노드는 헬로우 메시지를 수신하거나 연결 프로세스를 개시하여 헬로우 메시지를 수신하여 IG에 연결한다. 모든 트리-노드는 헬로우-타이머를 이용하여 주기적으로 그들의 부모 노드에 전송한다. 다른 이웃 노드는 헬로우 메시지를 수신함으로써 이웃 리스트를 업데이트시킬 수 있다. 트리-노드는 부모-변화 프로세스를 개시함으로써 그들의 부모 노드를 변경할 수 있으며, 자식-해제 프로세스를 사용하여 그들의 자식 노드를 해제할 수 있다. 다중 부모 노드를 가지는 쿼지-트리 방법과 달리, 모든 노드들은 하나의 부모 노드를 포함한다. 그러므로, 만일 트리-노드가 그들의 부모 노드와 링크가 해제되면, 고아 노드가 되며, 이에 따라, 연결 프로세스를 개시한다. 모든 노드는 그들의 트리 경로를 통해 IG에 주기적으로 등록한다. 본 시뮬레이션에서는 헬로우 타이머는 3-s로 설정한다.

[0105] 도 11a 내지 도 11c는 본 발명의 일 실시 예에 따른 기본 전개 시나리오를 설명하기 위한 도면들이다.

[0106] 도 11a 내지 도 11c에 도시된 바와 같이, 기본 시나리오(S1,S2, S3)에서 IG는 가운데, 가운데 상부, 가장자리에 IG를 위치되며, 시뮬레이션에서 트리 사이즈의 변화를 반영하기 위해 선택된다. 이는 종래의 접근 방법에서 사용하는 플러딩이 MNs와 IG 사이의 거리 및 트리 사이즈가 영향을 받으며, 차례로 트리 유지 오버헤드에 영향을 받기 때문이다.

[0107] 모바일 노드(nNodes)의 수는 25에서 100까지 변하며, 최대 속도(mSpeed)는 0에서 25까지 변한다. 다른 시뮬레이션 파라미터 및 값은 표 3에 도시된 바와 같다.

표 3

[0108]

Parameter	Value
Mobility pattern	Random waypoint
Pause time	30
Number of nodes	Varying density (1 fixed IG)
Dimension	1,000 × 1,000 m^2
Transmission range	250 m
Wireless bandwidth	2 Mbps
Simulation time	600 s (all sessions are alive until the end of the simulation)
Registration interval	5 s
Advertisement interval	2 s
Advertisement zone (IG)	2 hops
Increment in the expanding ring search (MN)	2 hops

[0109] 본 시뮬레이션에서는, 제어 오버헤드, 등록률, 등록 지연 및 등록 지터를 프로액티브, 리액티브, 하이브리드, TB, QT, QT-LDR에서 비교한다. 제어 오버헤드는 노드의 수에 의해 트리 유지와 노드 등록에 관련된 총 제어 오보헤드를 평균하여 얻어진다. 등록률은 아래의 수학식 2과 같이 등록 시도에 대한 성공률을 의미한다.

수학식 2

$$\text{Registration ratio} = \frac{\sum_{i=1}^{\text{nNodes}} \text{registration success of node } i}{\sum_{i=1}^{\text{nNodes}} \text{registration attempt of node } i}$$

[0110]

[0111] 이러한 메트릭은 패킷 전송량을 평가할 수 있다. 등록 지연은 MN에 의해 전송된 REG 메시지가 IG에 도달한 경과 시간이다. 등록 지터는 IG에 연속적으로 전송된 2개의 패킷의 평균 지연 시간의 차이를 의미한다. 이러한 메트릭들은 경로의 안정성을 평가할 수 있다.

[0112] 표 4는 제어 오버헤드에 영향을 미치는 중요한 기능들을 처리하는 방법을 나타낸다.

표 4

Functions	TB	QT	QT-LDR	Conventional approaches
Flooding	No	No	No	Yes
Broadcasting	Yes	No[a]	No[a]	Yes
Multi-path	No	Yes	Yes	No
Registration	Implicit	Implicit	Implicit	Explicit
Loop solution	No	Reduced loop possibility	Loop detection and resolution	N/A (no loop)

[0113]

[0114] QT 방법은 플러딩과 메시지 상관을 증가시키는 브로드캐스팅을 제거한다. 게다가, QT에서 노드는 다중 부모 노드를 관리하며, 이에 따라, IG에 다중 업스트림 경로를 유지할 수 있다. 이는 노드가 부모 노드와의 연결이 해제되어도 다른 부모 노드를 선택해서 제어 오버헤드를 줄일 수 있다. 또한, 자식 노드들 중 하나에 연결하기 위한 노드 확률을 줄일 수 있다. QT-LDR는 루프를 빨리 감지하고 분해할 뿐만 아니라 루프 형성 확률을 감소시킨다. 트리 기반 접근에서 이동성 관리(즉, 등록)에서, 루트(IG)가 전체 트리의 노드에 대한 정보를 관리하기 때문에, MN은 IG에 명쾌하게 등록하지 않는다.

[0115] 본 시뮬레이션에서는, 토폴로지 관리의 효율성에 대한 3가지 타입 프로토콜을 비교한다. 표 5는 본 시뮬레이션을 통해 토폴로지 관리에 대한 제어 메시지의 수를 나타낸다.

표 5

Si	Method	Topology management overhead					
		# of messages					Avg. KB for all msgs
		Hello	J-REQ	CR-REQ	REG (I)	REG (R)	
S1	TB	10,396	1,699	48	6,030	18,256	7.9
	QT	10,343	1,222	71	5,973	4,069	3.5
	QT-LDR	10,349	1,221	71	5,978	4,087	3.5
S2	TB	10,350	2,765	160	5,993	68,163	23.2
	QT	10,231	2,137	282	5,895	11,952	5.9
	QT-LDR	10,347	2,126	261	5,914	11,238	5.7
S3	TB	10,221	5,629	302	5,891	424,411	132.3
	QT	10,019	3,665	858	5,797	24,506	9.8
	QT-LDR	10,211	3,507	685	5,800	15,645	7.0

[0116]

[0117] 다중 부모 노드를 관리하는 다른 2가지 방법은 주 부모 노드와 링크가 해제된 노드를 위해 예비 부모 노드를 마련하기 때문에, TB에서 J-REQs가 가장 크다.

[0118] QT 및 QT-LDR은 MN이 링크된 자식 집합을 갖도록 하고, 더 많은 자식들을 관리하도록 하기 때문에 더 큰 CR-REQs을 나타낸다.

[0119] 한편, 가장 많은 REG(R)s은 루프가 형성되었다는 것을 나타낸다. TB는 QT와 비교하여 극도로 많은 수의 REG(R)를 나타내고, 루프 형성 가능성을 감소하고, QT-LDR와 비교할 때 형성하는 루프를 검출하고 분해한다. 또한, QT는 평균에서 더 큰 트리를 갖도록 S3에서 일부 루프를 생성한다.

[0120] 이하에서는, 쿼지 트리(Quasi-tree) 및 종래 방법을 비교 설명하도록 한다.

[0121] 플러딩을 이용하는 종래 방법은 MN으로부터 IG까지의 홉 거리가 비교적 짧을 경우 더 잘 동작한다. 종래 방법을 선호하는 시나리오 S1을 선택함에 의해 QT의 효과를 평가한다. 또한, QT와 TB는 비교하여 다중 부모의 유지 효과를 평가한다.

[0122] 도 12는 TB가 QT보다 좀더 많은 J-REQs를 생성하고, 디스패리티(disparity)가 mSpeed로 증가함을 보여준다. QT가 이용되는 경우 원 부모와 링크가 해제된 트리 노드는 MN-NN Join 프로세스를 통하지 않고, 다중 부모들 중 원 부모를 선택할 수 있다.

[0123] 도 12는 TB 및 QT의 mSpeed를 비교하기 위한 도면이다.

[0124] 도 12에 따르면, 다중 부모의 관리가 등록 경로의 안정성을 증가시키는 측면에서 매우 효과적이라는 점을 보여준다.

[0125] 도 13 내지 16은 다양한 시나리오에 따라 다양한 이동성 관리 프로토콜을 비교하기 위한 도면이다.

[0126] 프로액티브 방법은 낮은 오버헤드, 낮은 등록 지연시간(registration latency), 낮은 등록 지터(registration jitter)를 갖는다. 이러한 특징들은 기 정의된 경로를 따라 등록하기 위한 노드 시도를 자연스럽게 따라가게 된다. 하지만, 이러한 방법은 광고 메시지를 수신한 후에 등록의 즉각적인 개시 대신, MN이 모든 등록의 시작점에서 등록을 수행하기 때문에 최악의 등록 비율을 갖는다.

[0127] 결과적으로, 등록 경로는 다음 등록 인터벌 전에 무효가 될 수 있다. 프로액티브 방법과 대조적으로 리액티브 방법은 등록 경로를 설정한 후에 즉각적인 등록을 수행하도록 MN에 요청한다. 하지만, 이러한 방법은 다른 메트릭에 대해 높은 값을 나타낸다. 하이브리드 방법은 상술한 두 가지 접근의 단점을 모두 포함한다.

[0128] TB는 루프의 발행에 기인하여 좋은 성능을 갖지 않고, 실질적으로, S3에서 악화된다. 대조적으로 QT는 S3에서 등록 비율 그래프 패턴을 소량 감소시키고, 등록 지연시간 및 지터의 그래프 패턴을 소량 증가시키지만, 모든 메트릭에 대해 훌륭한 결과를 보여주고, 증가된 트리의 평균 사이즈 때문에 루프를 생성할 가능성이 높다.

[0129] 도 17 내지 20은 nNode가 가변함에 따른 프로토콜을 비교하는 도면들이다.

[0130] 구체적으로, 도 17 내지 20은 nNodes (노드 수)가 25에서 100까지 가변함에 따른 프로토콜을 비교하는 도면들이다.

[0131] 프로액티브의 제어 오버헤드는 IG가 주기적으로 네트워크와 ADV 메시지를 플러딩하기 때문에 nNode의 증가에 민감하지 않다. 하지만, 리액티브의 제어 오버헤드는 모든 노드들이 기존 경로가 무효화될 때마다 SOL 메시지를 플러딩하기 때문에 nNode의 증가에 민감하다.

[0132] 리액티브 프로토콜에서 등록 지연시간 및 등록 지터는 nNode의 증가에 더 민감하다. 또한, TB 에서 nNode의 증가에 따라 루프 형성의 증가된 가능성은 제어 오버헤드의 패턴 증가를 가져온다. 25 노드를 갖는 등록 비율은 50 노드를 갖는 등록 비율 보다 다소 낮다. 이는 전자가 네트워크의 파티션을 더 많이 경험하기 때문이다.

[0133] 대조적으로 QT는 노드가 루프 형성 가능성을 낮추고, 업스트림 경로를 지속할 확률을 높이는데 기여하는 더 많은 부모들을 유지할 수 있고, 노드의 개수가 증가함에 따라 더 안정적이 된다. QT는 노드 개수에 대항하여 좋은 확장성을 갖는다.

[0134] 도 21 내지 24는 mSpeed의 가변에 따른 프로토콜을 비교하는 도면들이다.

[0135] mSpeed에서 증가에 대항한 TB의 성능 악화는 파괴된 링크의 작은 수리에 기인한다. 하지만, QT는 S1에서 MN이

파괴된 링크의 빠른 수리를 가능하게 하는 더 많은 부모를 갖는 경향이 있기 때문에 두드러진 성능 악화를 가져오지는 않는다.

[0136] 지금까지의 시뮬레이션 결과에 따라, QT는 다양한 네트워크 전개 시나리오 및 노드 밀집도 및 노드 이동성의 상이한 레벨에 대항하여 매우 의존적이고 확장적임을 알 수 있다.

[0137] 이하에서는 루프 검출 및 분해 알고리즘을 평가하도록 한다.

[0138] LDR 알고리즘을 좀 더 효율적으로 평가하기 위하여 더 큰 트리에서 결과를 유도하여 루프 형성 확률을 증가시키는 시나리오 S3를 사용한다. 또한, 본 시뮬레이션은 높은 노드 이동성이 토폴로지 변화 및 그에 따른 루프 형성 가능성을 증가시키기 때문에 넓은 범위의 노드 스피드를 이용한다. 본 시뮬레이션 시나리오는 증가된 트리 사이즈 및 높은 노드 이동성에 대항하여 QT-LDR의 확장성을 제공하는데 기여한다.

[0139] 도 25 내지 28은 본 발명의 일 실시 예에 따른 시나리오에서 QT 및 QT-LDR을 비교하는 도면들이다.

[0140] 구체적으로, 도 25 내지 28은 50 개의 nNodes 및 0에서 50 m/s로 가변하는 mSpeed를 갖는 S3에서 QT 및 QT-LDR을 비교한다.

[0141] 도 25는 QT가 10 m/s보다 큰 mSpeed에 대해 제어 오버헤드를 가파르게 증가시킨다는 점을 보여준다. 이는 높은 노드 이동성이 링크 파괴 확률 및 그에 따른 루프 형성 확률을 증가시키기 때문이다. 하지만, QT-LDR의 제어 오버헤드는 50 m/s에서 오버헤드가 0m/s에서의 제어 오버헤드의 두 배라는 점을 제외하고 mSpeed에서의 가변에 매우 민감하지 않다.

[0142] 도 26에 따르면, 등록 비율은 제어 오버헤드에 역비례한다. 또한, 도 27 및 28에서 QT의 등록 지연시간 및 지터는 루트 형성의 빈도에 대해 가파르게 증가한다. 하지만, QT-LDR이 효과적으로 루프를 분해한다는 점이 도시된 바로부터 추정될 수 있다. 결과적으로, LDR은 분명히 효과적이고, QT-LDR 이동성 관리 프로토콜은 큰 트리 및 높은 노드 이동성에 대항하여 양호한 확장성을 보여준다.

[0143] 본 시뮬레이션 결과는 QT의 다중 부모 관리가 루프 형성 가능성을 감소시킬지라도 QT가 루프를 완전히 분해하지 않는다는 점을 보여준다. 또한, 본 시뮬레이션 결과는 QT-LDR이 더 큰 트리 및 높은 노드 이동성에 대해 매우 강인하다는 점을 보여준다.

[0144] 도 29는 본 발명의 일 실시 예에 따른 트리 형성 방법을 설명하기 위한 흐름도이다.

[0145] 도 29에 도시된 이동성 애드 혹 네트워크(Mobile ad hoc networks : MANETs)에서의 트리 형성 방법에 따르면, 우선, 인터넷 게이트웨이(internet gateway)에 연결된 복수의 노드 및 복수의 노드 각각에 순차적으로 연결된 적어도 하나의 노드를 마련한다(S2910).

[0146] 이어서, 적어도 하나의 노드 중 제1 노드가 자신의 부모 노드로 유니캐스트한 메시지를, 제1 노드의 인접 노드인 제2 노드가 수신한다(S2920).

[0147] 이어서, S2920 단계에서 제2 노드가 수신된 메시지에 기초하여 자신의 부모 노드와 인터넷 게이트웨이 사이의 제1 링크 수 및 제1 노드와 인터넷 게이트웨이 사이의 제2 링크 수를 비교한다(S2930).

[0148] 이 후, S2930 단계의 비교 결과에 따라 제2 노드가 제1 노드를 자신의 예비 부모 노드로 추가 또는 자신의 부모 노드로 설정한다(S2940).

[0149] 또한, S2940 단계에서는, S2930 단계의 비교 결과, 제1 링크 수 및 제2 링크 수가 동일하면 제2 노드가 제1 노드를 자신의 예비 부모 노드로 추가할 수 있다.

[0150] 또한, S2940 단계에서는, S2930 단계의 비교 결과, 제1 링크 수가 제2 링크 수보다 크면, 제1 노드를 부모 노드로 설정하고 자신의 부모 노드와 연결을 해제할 수 있다.

[0151] 또한, S2910 단계에서는 복수의 노드 각각이 인터넷 게이트웨이에 의해 브로드캐스트된 메시지를 수신하고, 수신된 메시지에 응답하여 링크 연결 요청 메시지를 인터넷 게이트웨이에 전송하고, 링크 연결 요청 메시지에 대한 응답으로 확인 메시지를 수신하면, 복수의 노드 각각이 인터넷 게이트웨이에 연결될 수 있다.

[0152] 또한, 제3 노드가, 복수의 노드 중 하나가 인터넷 게이트웨이에 유니캐스트한 메시지를 수신하고 수신된 메시지에 응답하여 링크 연결 요청 메시지를 복수의 노드 중 하나에 전송하고, 링크 연결 요청 메시지에 대한 응답으로 확인 메시지가 수신되면, 제3 노드는 복수의 노드 중 하나에 연결될 수 있다.

[0153] 또한, 제2 노드가 자신의 부모 노드와 연결이 해제되고, 자신의 자식 노드로부터 메시지를 수신하고, 메시지를 수신한 제2 노드가 자신의 부모 노드로 연결 해제 메시지를 전송하고, 연결 해제 메시지에 대응하여, 제2 노드의 자식 노드가 새로운 노드를 부모 노드로 설정할 수 있다.

[0154] 또한, 제2 노드에 연결된 고아 노드가 고아 노드와 인터넷 게이트웨이 사이의 링크 수를 변경하고, 고아 노드가 자식 노드를 갖는 경우 제2 노드에 메시지를 전송할 수 있다. 이어서, 메시지를 수신한 자식 노드가 자식 노드와 인터넷 게이트웨이 사이의 링크 수를 변경하고, 변경된 링크 수가 기 설정된 값을 초과하면, 자식 노드가 고아 노드와의 연결을 해제할 수 있다.

[0155] 상술한 바와 같이 본 발명에서는 MN이 가능하다면 다중 부모를 유지하고, 모든 부모 노드에 대한 링크가 해제될 때까지 IG로의 업스트림 경로를 유지하는 새로운 이동성 관리 방법을 제안한다. 루프는 노드가 유일한 부모 노드와 링크가 해제되는 경우 형성될 수 있다. 하지만, 루프는 MN 트랙을 가짐으로써 분해될 수 있고, IG 까지의 홉 거리를 제한할 수 있고, G 까지의 홉 거리가 변화됨에 따라 MN을 가짐으로써 헬로우 메시지를 즉시 전송할 수 있다.

[0156] QT 및 QT-LDR은 TB 및 간단한 종래의 접근 방법(Proactive, Reactive, Hybrid)과 비교되었다. 시뮬레이션 결과는 QT-LDR 방법인 큰 트리 및 높은 노드 이동성에 대항하여 확장성이 높다는 점을 보여준다. 또한, QT는 특정 노드에 트래픽의 바이어스를 향상시키도록 용이하게 확장될 수 있고, 노드는 각 부모에 의해 발생된 패킷 개수를 카운팅하는 간단한 카운팅 기술을 이용하여 부모의 로드를 평가한다. 이 경우 노드는 패킷을 포워딩하기 전에 부모 노드들 중 적게 로딩된 노드를 선택할 수 있게 된다.

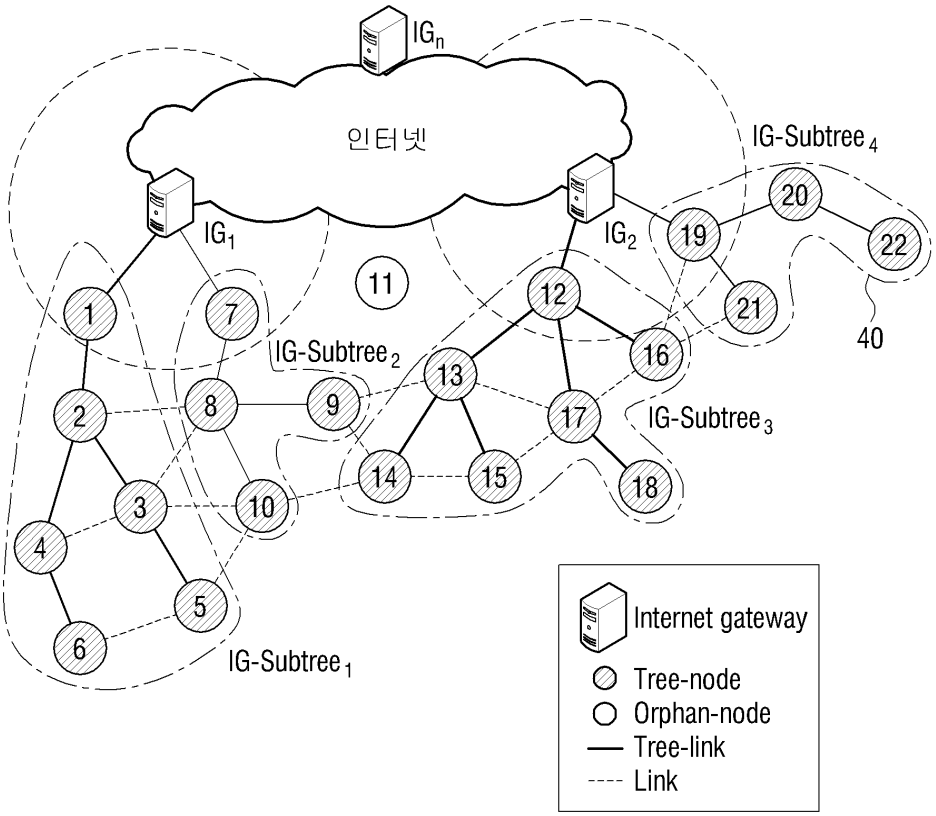
[0157] 또한, 이상에서는 본 발명의 바람직한 실시 예에 대하여 도시하고 설명하였지만, 본 발명은 상술한 특정의 실시 예에 한정되지 아니하며, 청구범위에서 청구하는 본 발명의 요지를 벗어남이 없이 당해 발명이 속하는 기술분야에서 통상의 지식을 가진 자에 의해 다양한 변형실시가 가능한 것은 물론이고, 이러한 변형실시들은 본 발명의 기술적 사상이나 전망으로부터 이해되어져서는 안 될 것이다.

부호의 설명

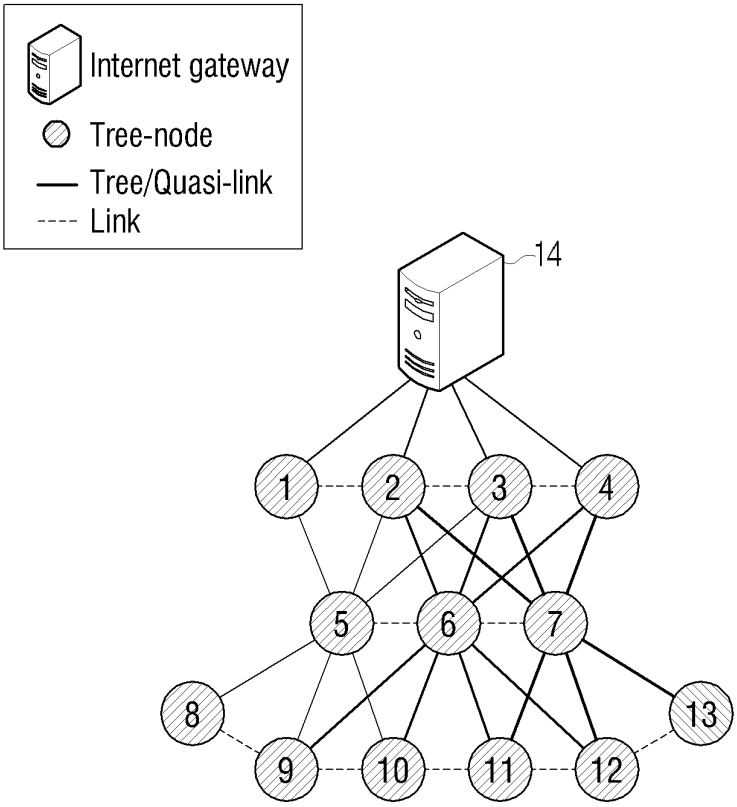
[0158] 1 내지 13 : 트리 노드 14 : 인터넷 게이트웨이

도면

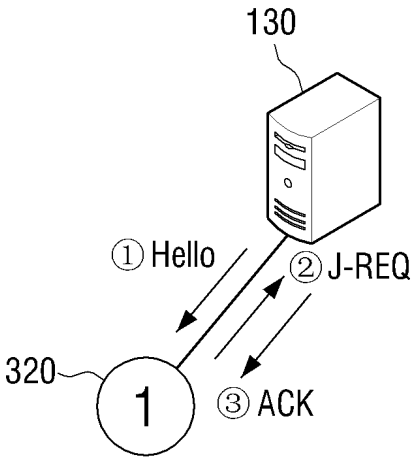
도면1


IG_n
인터넷
IG_1
IG_2
IG-Subtree_4
IG-Subtree_2
IG-Subtree_3
IG-Subtree_1
40
Internet gateway
Tree-node
Orphan-node
Tree-link
Link

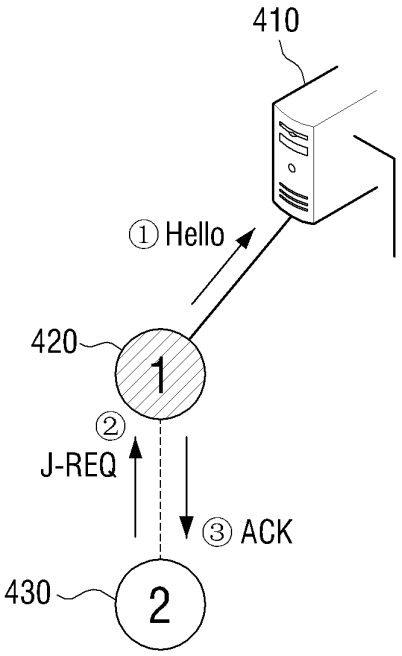
도면2


Internet gateway
Tree-node
Tree/Quasi-link
Link
14
1
2
3
4
5
6
7
8
9
10
11
12
13

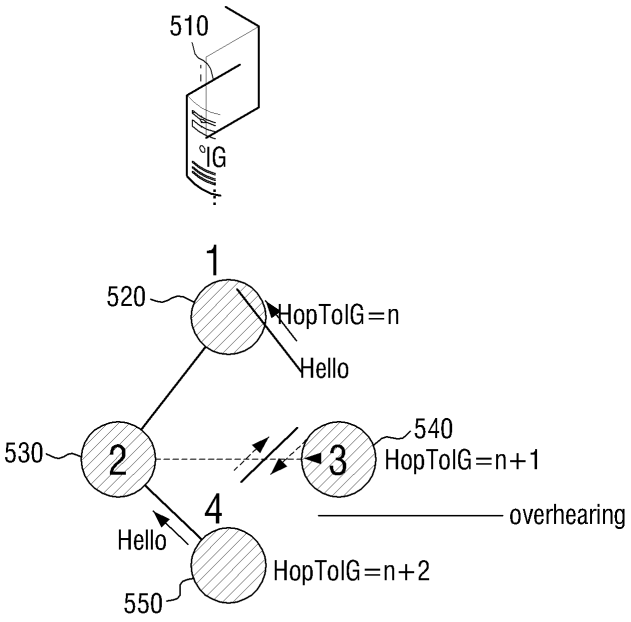
도면3


130
① Hello
② J-REQ
③ ACK
320
1

도면4


410
① Hello
420
1
②
J-REQ
③ ACK
430
2

도면5


510
IG
1
520
HopToIG=n
Hello
530
2
3
540
HopToIG=n+1
overhearing
4
Hello
HopToIG=n+2
550

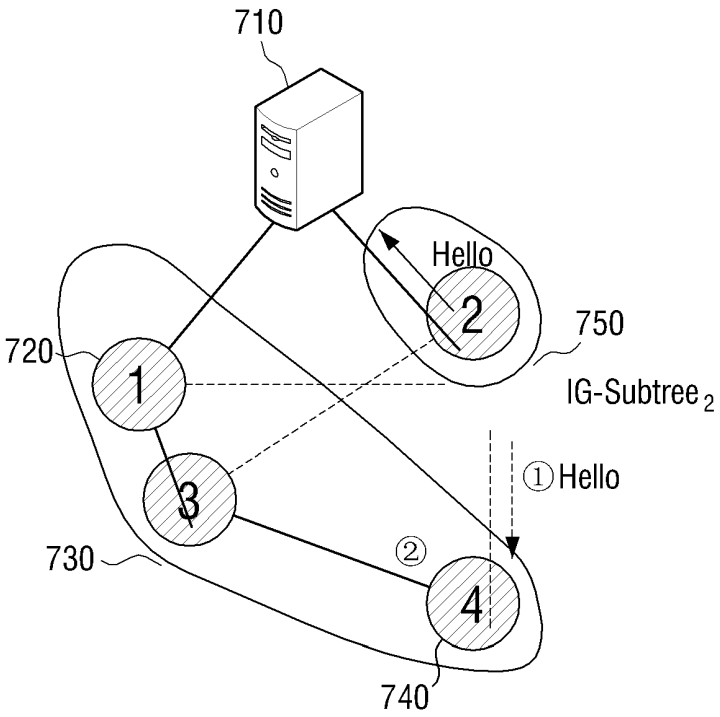
도면6

```
// Hx : node x's HopToIG
// x.P: A set of x's parents;   x.C: a set of x's children
At a node x that overhears Hello from one of its parent y:
IF node y is a primary parent THEN
    IF (Hx == Hy + 1) THEN // no change
        x resets the timer for parent y;
    ELSE IF (Hx < Hy + 1) and   ((x.P  – {y}) != ∅) THEN
        x selects its primary parent from (x.P – {y});
    ELSE // parent y has reduced its HopToIG or no parent to select
        Hx = Hy + 1;
        x sends Hello to its primary parent y;
    ENDIF;
ELSE
    IF Hx == Hy + 1 THEN
        x resets the timer of parent y;
    ELSE IF Hx > Hy + 1 THEN    // y has reduced its HopToIG
        x changes its primary parent to y;
    ELSE
        x.P = x.P – {y};
    ENDIF;
ENDIF;

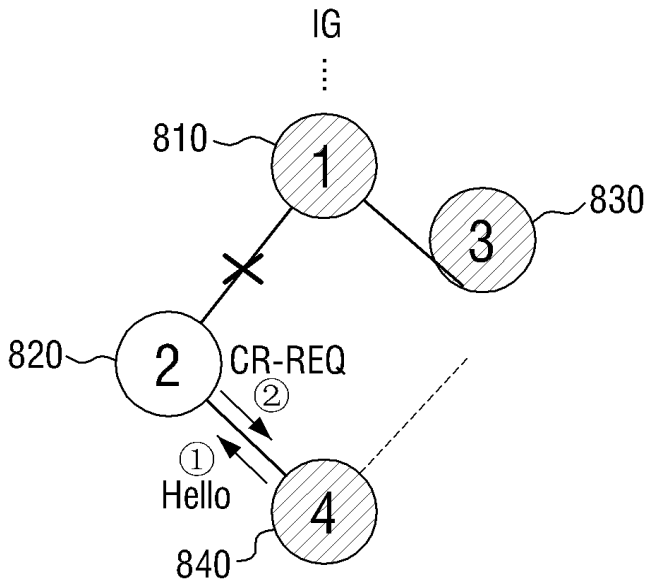
At a node x that receives Hello from y in x.C:
    x resets the timer for child y;

At a node x that overhears Hello from y in x.C:
IF (Hx == Hy + 1) THEN
    x resets the timer for child y;
ELSE
    x.C = x.C  – {y};
ENDIF;
```

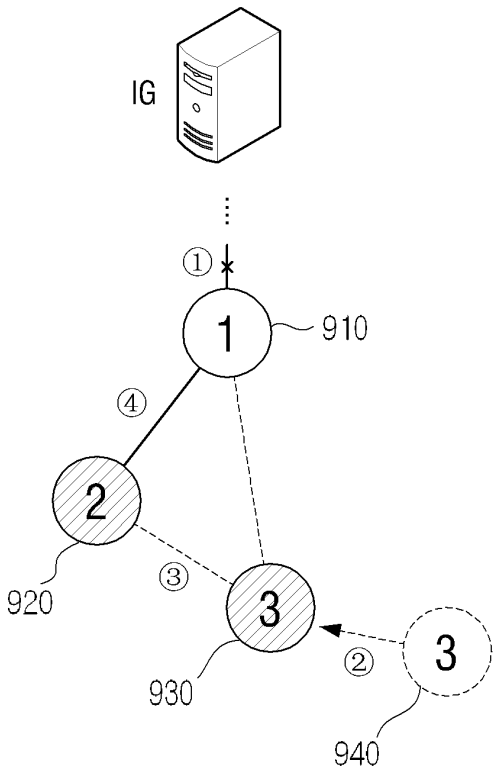
도면7



도면8


IG
810
1
830
3
820
2
CR-REQ
②
①
Hello
4
840

도면9

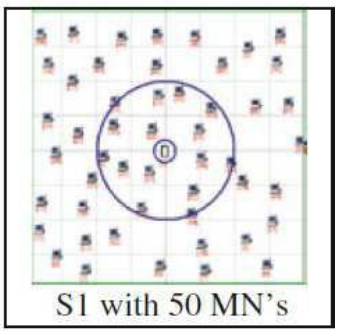

IG
①
1
910
④
2
920
③
3
930
②
3
940

도면10

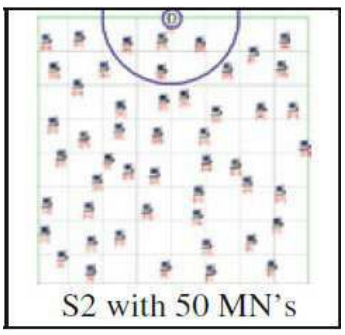
```
// H_x : The HopToIG of node x
// x.PP: The primary parent of node x
At a node x that detects disconnection to its only parent
IF node x overhears Hello from a tree-node y THEN
    IF H_y == DEPTH_LIMIT THEN    return; ENDIF;
    // x is not an ancestor of node y or
    // x may not have received update message (REG) initiated by y
    IF x's descendant list does not include y THEN
        x joins y;
        H_x = H_y + 1;
        x immediately sends Hello to y if x.C != ∅;
    ENDIF;
ENDIF;

At a node x that overhears Hello from its primary parent x.PP
//assume that node x has one parent x.PP only
IF H_x.PP == DEPTH_LIMIT THEN
    Node x breaks the loop by becoming an orphan-node;
ELSE
    IF (H_x != H_x.PP + 1) THEN
        H_x = H_x.PP + 1;
        x immediately sends Hello to x.PP if x.C != ∅;
    ENDIF;
ENDIF;
```

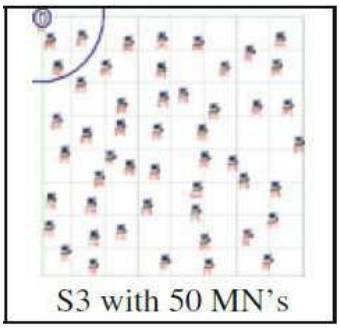
도면11a



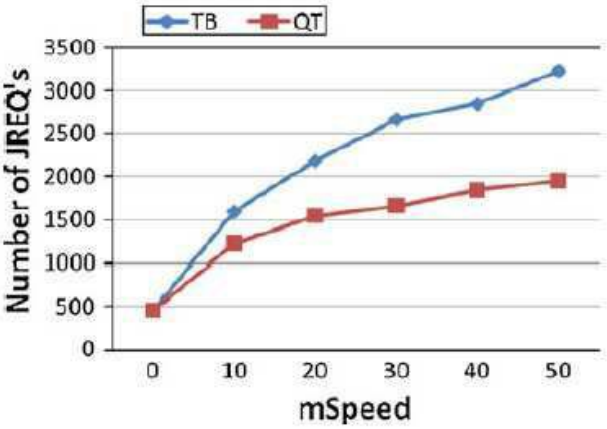
도면11b



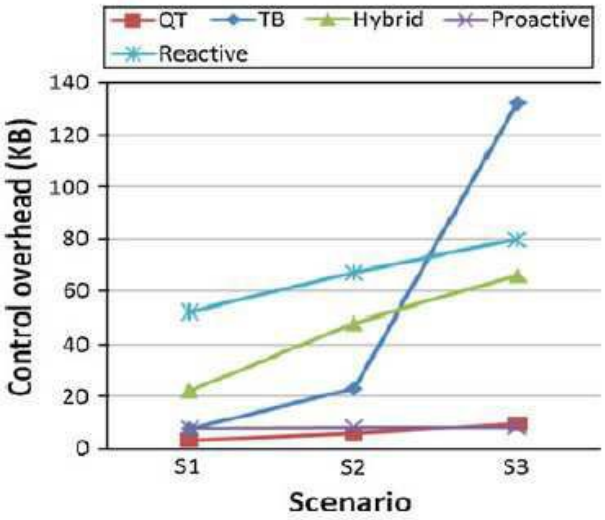
도면11c



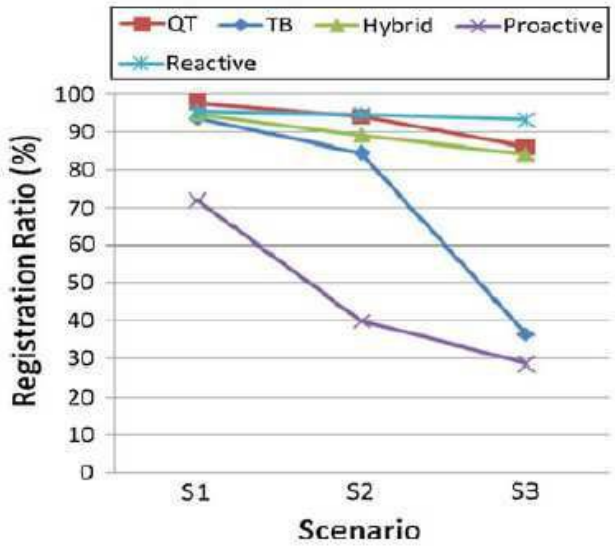
도면12



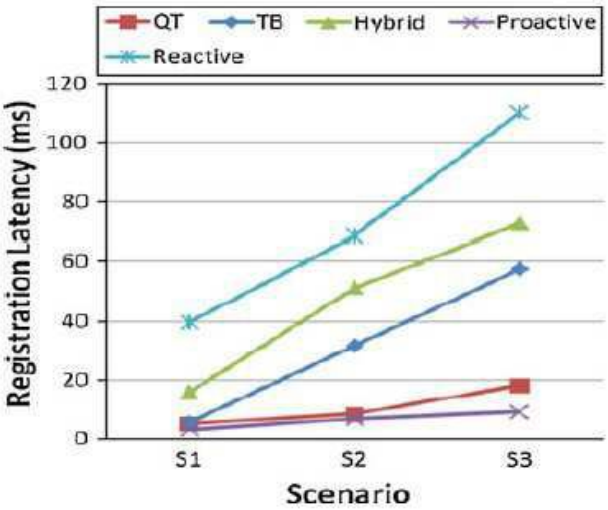
도면13



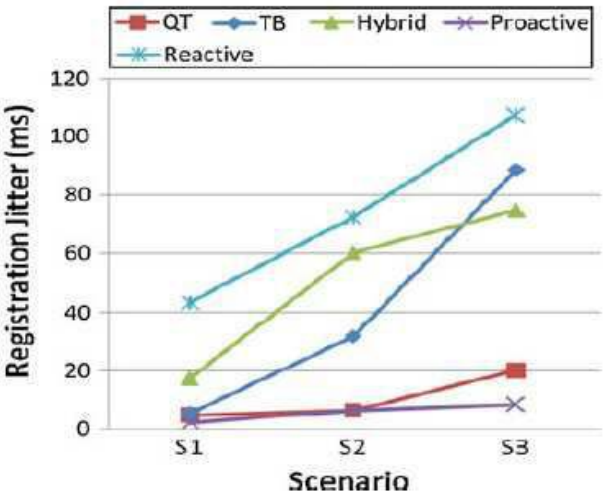
도면14



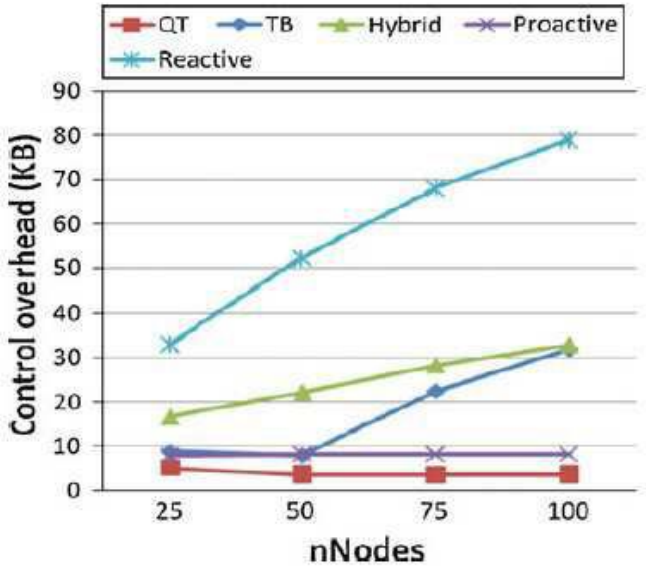
도면15



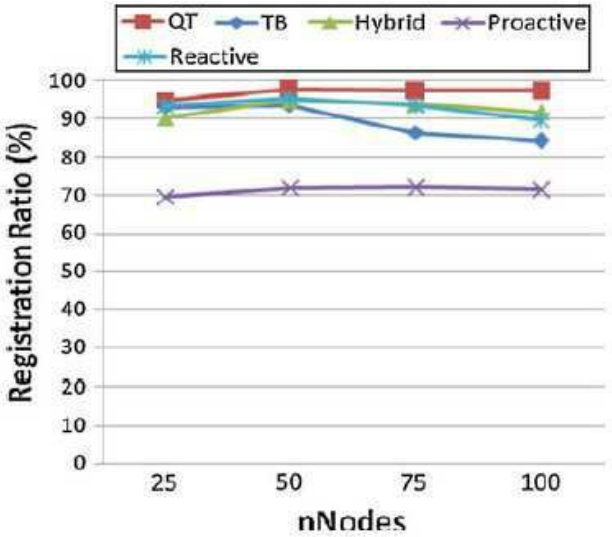
도면16



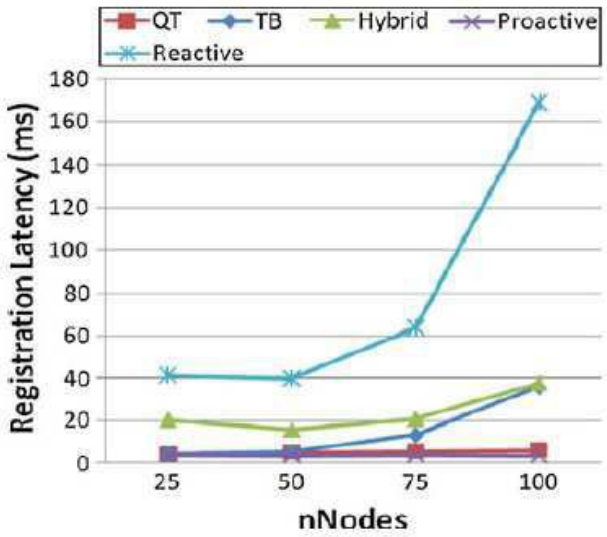
도면17


QT
TB
Hybrid
Proactive
Reactive
Control overhead (KB)
90
80
70
60
50
40
30
20
10
0
25
50
75
100
nNodes

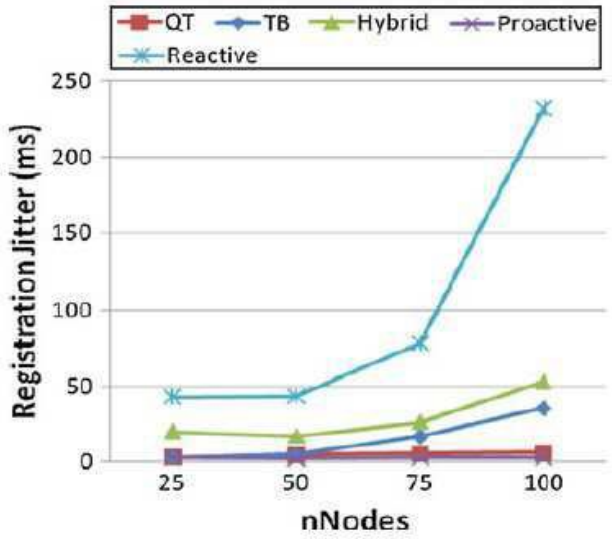
도면18


QT
TB
Hybrid
Proactive
Reactive
Registration Ratio (%)
100
90
80
70
60
50
40
30
20
10
0
25
50
75
100
nNodes

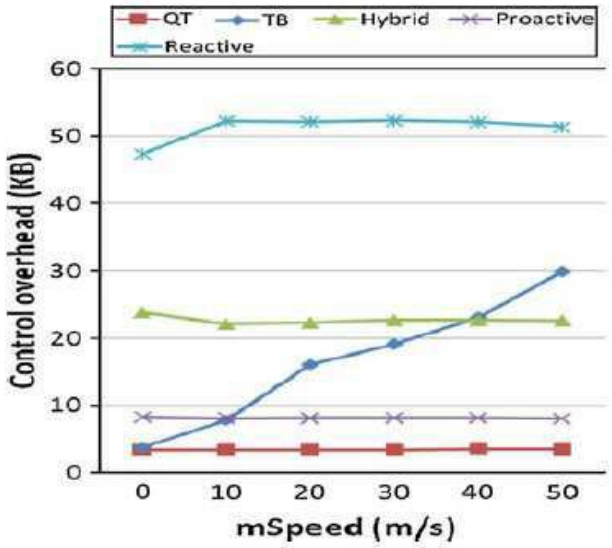
도면19


QT
TB
Hybrid
Proactive
Reactive
Registration Latency (ms)
180
160
140
120
100
80
60
40
20
0
25
50
75
100
nNodes

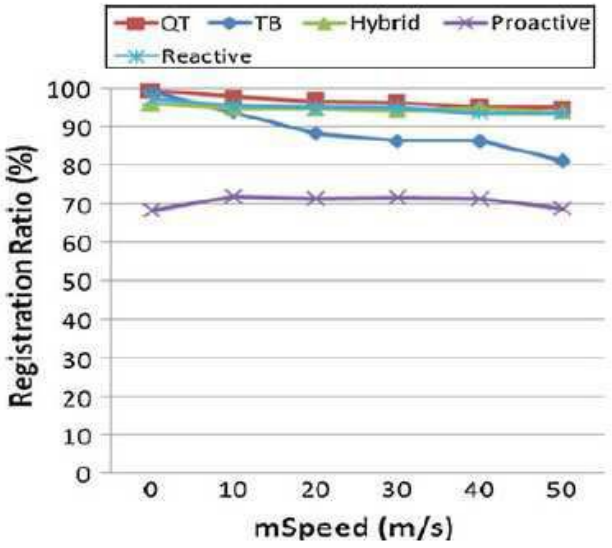
도면20


QT
TB
Hybrid
Proactive
Reactive
Registration Jitter (ms)
250
200
150
100
50
0
25
50
75
100
nNodes

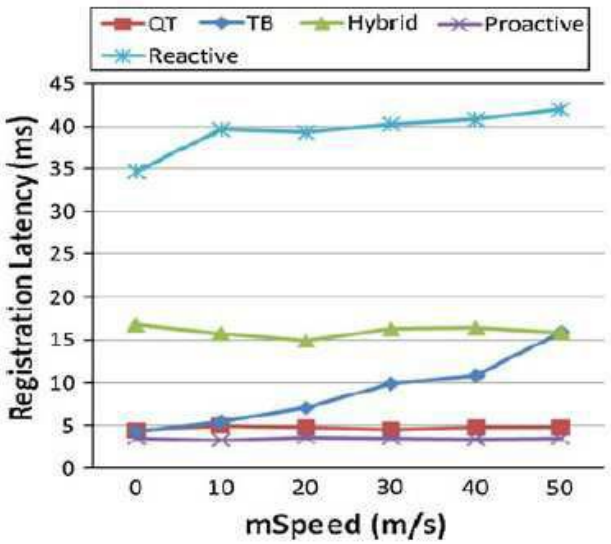
도면21


QT
TB
Hybrid
Proactive
Reactive
Control overhead (KB)
60
50
40
30
20
10
0
0
10
20
30
40
50
mSpeed (m/s)

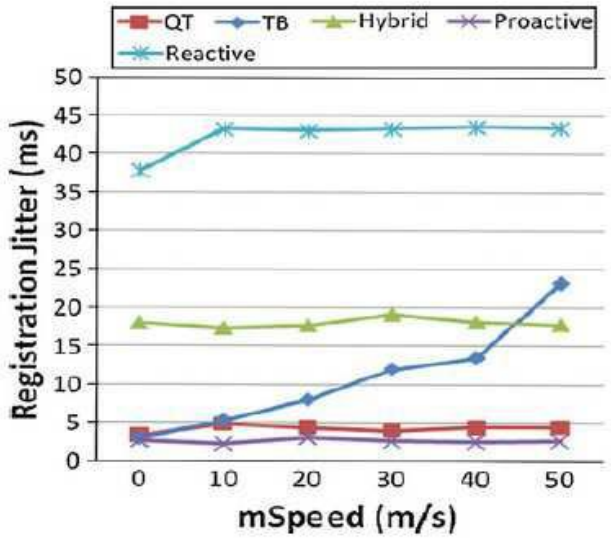
도면22


QT
TB
Hybrid
Proactive
Reactive
Registration Ratio (%)
100
90
80
70
60
50
40
30
20
10
0
0
10
20
30
40
50
mSpeed (m/s)

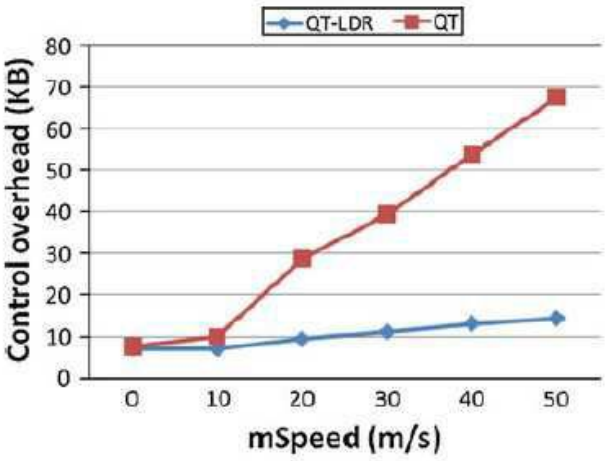
도면23



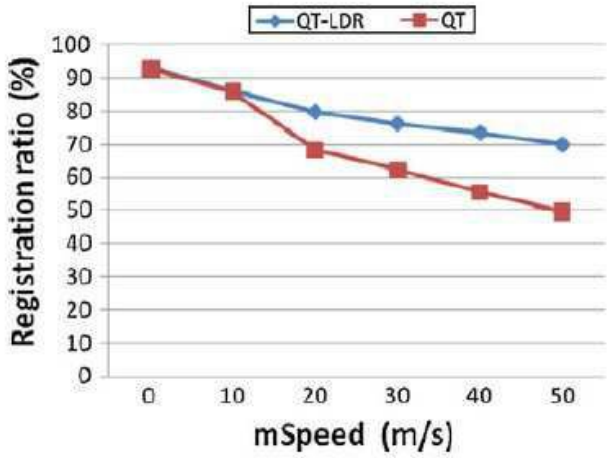
도면24



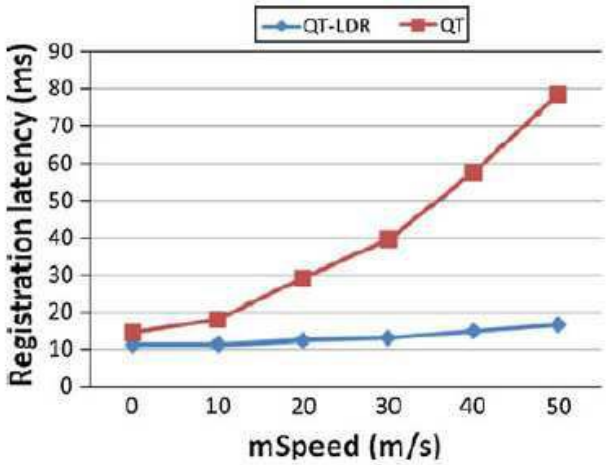
도면25



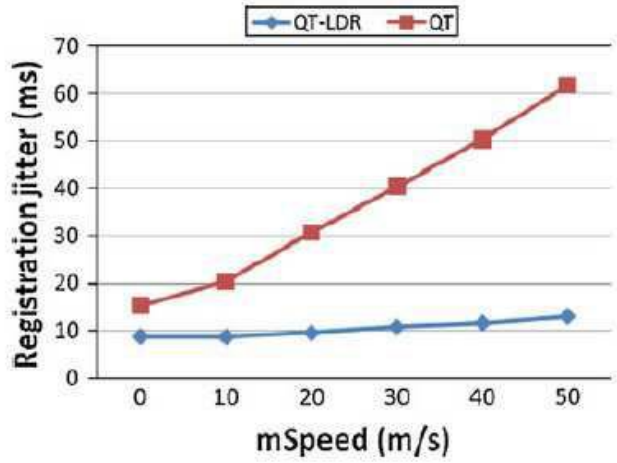
도면26



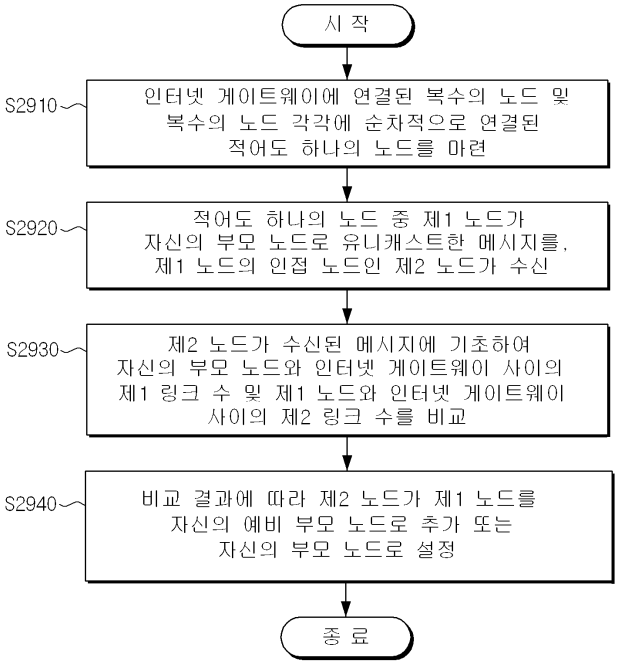
도면27



도면28



도면29


시 작
S2910
인터넷 게이트웨이에 연결된 복수의 노드 및 복수의 노드 각각에 순차적으로 연결된 적어도 하나의 노드를 마련
S2920
적어도 하나의 노드 중 제1 노드가 자신의 부모 노드로 유니캐스트한 메시지를, 제1 노드의 인접 노드인 제2 노드가 수신
S2930
제2 노드가 수신된 메시지에 기초하여 자신의 부모 노드와 인터넷 게이트웨이 사이의 제1 링크 수 및 제1 노드와 인터넷 게이트웨이 사이의 제2 링크 수를 비교
S2940
비교 결과에 따라 제2 노드가 제1 노드를 자신의 예비 부모 노드로 추가 또는 자신의 부모 노드로 설정
종 료