한국ITS학회논문지
제8권, 제1호 (2009년 2월)
pp. 65~75

VANET환경에서의 효율적인 데이터 접근성 향상기법

Efficient Method for Improving Data Accessibility in VANET

심 규 선* 이 명 수** 이상근***
(Kyu-Sun Shim) (Myong-Soo Lee) (Sang-Keun Lee)

요 약

차량 애드 혹 네트웍 (VANET)은 모바일 애드혹 네트워크의 일종으로써 차량 간의 임시적인 통신을 가능케 한다. VANET의 모바일 노드는 네트웍의 일원으로 참여하면서 에너지와 자원을 사용한다. 몇몇의 노드는 다른 노드와 협력하는 대신에 자신의 이득만을 취하려고 하는 이기성을 가지고 있다. 이러한 이기성을 가진 노드들로 인해 데이터 접근성과 네트웍의 효율성이 감소하게 된다. 이 논문에서는 새로운 기법인 VANET 그룹에서 이기성을 갖고있는 노드를 제거하는 Friendship-VaR을 제안한다. Friendship-VaR은 이기성을 갖고있는 노드를 제거하고 믿을 수 있는 노드들간에 데이터 공유를 가능케 한다. Friendship-VaR는 간단한 데이터 교환을 통해 노드의 이기성 여부를 결정한다. 실험 결과는 제안한 기법이 기존의 기법에 비해 데이터 접근성이 좋아짐을 보여준다.

Abstract

A Vehicular Ad-Hoc Network (VANET) is a form of Mobile ad-hoc network, to provide temporary communications among nearby vehicles. Mobile node of VANET consumes energy and resource with participating in the member of network. Some node tends to have a selfishness to place one's own profits above cooperation with others. As result of selfish node, it reduces data accessibility and the efficiency of networks. In this paper, we propose noble method, Friendship-VaR that excludes selfish nodes from a group of VANET. Friendship-VaR enables to improve data accessibility by eliminating selfish nodes and sharing data among reliable nodes. Friendship-VaR determines selfishness of nodes by simple data exchange. The experiments shows proposed method outperform existing method in terms of data accessibility.

Key words: VANET, selfishness, data replication, data accessibility, mobile computing

Ⅰ. 서 론

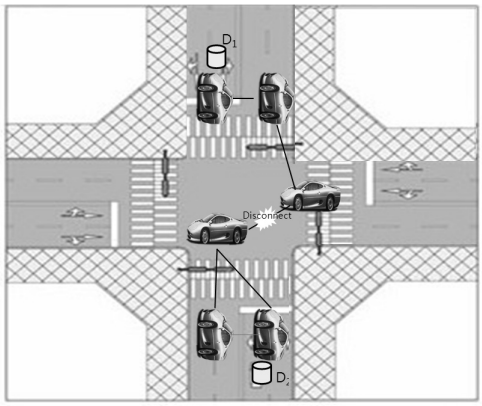
최근 지능형 차량 및 ITS (Intelligent Transportation System) 연구개발을 통해 차량에 IT기술을 접목시키기 위하여 많은 연구가 이루어지고 있다. 특히 국내의 경우 산업자원부가 미래 10대 전략 품목으로 자동차 부분에서 지능형 차량을 선정하였으며, 이러한 ITS기술의 한 분야로 움직이는 차량 간의 임시적인

† 본 연구에 참여한 연구자의 일부는 '2단계 BK21사업'의 지원비를 받았음.
* 주저자 : 고려대학교 컴퓨터학과 학부과정
** 공저자 : 고려대학교 컴퓨터학과 박사과정
*** 공저자 : 고려대학교 컴퓨터학과 부교수
† 논문접수일 : 2008년 12월 15일
† 논문심사일 : 2009년 2월 13일
† 게재확정일 : 2009년 2월 20일

통신을 가능하게 하는 VANET (Vehicular Ad-hoc Network)이 주목을 받고 있다 [1-4].

VANET은 MANET (Mobile Ad-hoc Network)의 한 분야로 차량 간의 통신 또는 차량과 노변 장치 간의 통신을 가능하게 하며 각 장치들은 애드혹 네트워크의 노드와 같은 역할을 한다. 애드혹 네트워크는 별도의 기반 시설 (Infrastructure)이 없어도 노드들 간의 무선 링크를 통해 연결되어 정보를 전달할 수 있는 정보 전송 네트워크를 말한다 [3]. 하지만 VANET에서는 차량이 이동하기 때문에 노드들 간의 연결이 빈번하게 끊긴다. 빈번한 이동은 안정되게 연결된 노드들이 이동하지 않은 채 연결되어 있는 안정된 네트워크보다 불안정한 정보 전송을 초래한다 [5].

애드혹 네트워크 (Ad-hoc Network)에서는 노드의 이동성으로 인해 네트워크가 빈번하게 끊어진다. 그렇기 때문에 다른 노드가 가지고 있는 데이터를 요청하여 가지고 올 수 있는 가능성인 데이터 접근성 (Data Accessibility)이 낮아지며, 애드혹 네트워크 환경에서의 데이터 접근성이 노드의 움직임이 없이 연결된 네트워크에 비해 현저하게 낮기 때문에 문제가 된다. 예를 들면 하나의 애드혹 네트워크가 노드가 이동함에 따라 두 개의 네트워크로 나누어지기 쉬우며, 나누어진 네트워크 간의 정보 전송이 이루어질 수가 없다. 그러면 나누어진 하나의 네트워크에 포함된 노드가 다른 네트워크에 속해있는 노드가 가진 데이터를 이용하려고 해도 접근할 수 없기 때문에 이용할 수 없다. <그림 1>을 보면 가운데 교차로를 건너고 있는 차량 간의 이동으로 인해 연결이 끊겨서 네트워크가 아래와 위 네트워크로 나누어진다. 이런 네트워크 분할로 인해서 각각 위 아래에 있는 D_1과 D_2는 서로 접근할 수 없게 된다. 이러한 연유로 기존 애드혹 네트워크에서는 데이터의 접근성을 향상시키는 문제가 중요한 이슈가 되고 있다.

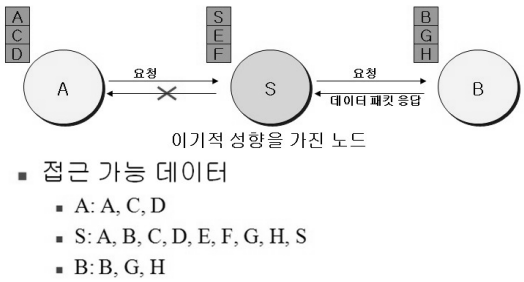


<그림 1> 네트워크 분할
<Fig. 1> Network partitioning

애드혹 네트워크에서 데이터 접근성을 향상시키기 위해서 다양한 방법이 제시되고 있으며, 많은 연구들이 진행되었다. 주요 연구 중에 하나는 데이터 복제본을 이용하는 방법이다 [6-8]. 이는 효율적인 데이터 복제 방법을 통해 네트워크가 분할되기 전에 미리 원본 데이터의 복제본을 복제하여 같은 네트워크 안의 노드끼리 공유함으로써 데이터의 접근성을 향상시킨다.

하지만 대부분의 움직이는 노드들은 자원이 제한되어 있어 다른 노드들을 위해 자신의 자원을 사용하기 보다는 자신의 이익을 위해서 사용하고자 하기 때문에 이기적인 성향이 생겨난다. 그리고 [9]에서 언급하였듯이, 실제 사용자간의 데이터를 공유하는 대부분의 P2P (Peer-to-Peer) 시스템에서 70%에 가까운 사용자들이 자신의 데이터를 공유하지 않고 다른 사용자로부터 데이터를 받고만 있다. 이런 상황은 애드혹 네트워크에서도 쉽게 적용되는데, 대부분의 움직이는 노드들은 자원이 제한되어 있기 때문에 자신의 자원을 사용하기 보다는 네트워크의 다른 노드들의 자원을 사용하여 자신의 자원을 아끼고자 한다. 이미 모바일 애드혹 네트워크에서 이기적인 성향을 가진 노드들에 대한 처리는 라우팅 단계에서 많은 연구가 진행되고 있으며, [10]에서는 한 네트워크 내의 약 40%의 노드가 이기적인 행동을 하는 것을 발견하였다.

기존 연구에서는 네트워크 내에 데이터의 복제본을 노드들끼리 공유함으로써 데이터 접근성을 향상시킨다. 하지만 데이터를 공유하는 네트워크 내에서 이기적인 성향을 가진 노드가 있다면 복제한 자신의 데이터를 다른 노드에게 전송하지 않고, 다른 노드들의 데이터만 받고자 하기 때문에 데이터의 접근성을 떨어진다. <그림 2>에서 보면, 노드 A, B, S는 하



<그림 2> 이기적인 노드의 영향
<Fig. 2> Effectiveness of selfish node

나의 네트워크에서 서로 데이터를 공유하고자 한다.

하지만 가운데 위치하는 S노드는 이기적인 성향이 있어 자신의 데이터를 다른 노드들에게 공유하지 않은 뿐더러 A와 B 사이의 데이터 패킷의 전송도 저해시키므로 데이터 접근성을 떨어뜨리는 원인이 된다. 이기적인 노드들에 의해서 데이터 접근성이 떨어지는 것을 막기 위해서는 <그림 2>와 같은 상황에서 S를 미리 찾아내는 것이 중요한 문제이다.

본 연구는 각 노드들이 가지고 있는 자원의 정보를 이용하여 이기적인 노드를 찾아내는 기법인 Friendship-VaR를 제안한다. 이 기법은 애드혹 환경에서 데이터 공유 그룹을 만들기 전에 미리 이기적인 노드를 찾아내어 공유그룹에서 제외하는 것을 목적으로 한다. 이 기법을 이용하면 노드 간의 데이터 접근성을 향상시킬 뿐만 아니라 네트워크의 효율 역시 증가하여 우수한 성능을 보여준다.

본 논문의 구성은 다음과 같다. 우선 2장에서는 본 연구와 관련된 연구에 관하여 설명하고, 3장에서는 본 논문에서 제안하는 기법에 대해 설명한다.

Ⅱ. 관 련 연 구

이 장에서는 모바일 애드혹 네트워크 환경에서 복제본 할당을 통해 데이터 접근성을 향상시키는 방법과 기존의 라우팅과 연관되어 정의된 이기적인 노드에 대한 정의, 그리고 금융학에서 위기를 측정하기 위한 모델인 VaR (Value at Risk)에 대해 설명한다.

1. 복제본 할당 기법

복제본 할당 기법은 모바일 애드혹 네트워크 환경에서 데이터 접근성을 높이기 위해서 고안된 방법으로 T. Hara가 제안한 기법인 SAF, DAFN, DCG 등이 대표적이다 [8]. 이 방법들은 애드혹 환경에서 움직이는 노드의 이동성에 의해 낮아지는 데이터 접근성을 향상시키고자 한다.

SAF (Static Access Frequency)는 모든 노드들이 데이터 요청 우선순위를 각 데이터에 대한 접근 빈도에 따라 정렬하고, 정렬된 데이터의 접근 빈도에 따라 노드들은 요청 우선순위가 높은 데이터의 복제본을 할당받는 방식이다. 하지만 노드들이 각 데이터에 대해 접근 빈도가 비슷하다면 우선순위가 비슷해지기 때문에 노드들이 할당받는 데이터 복제본이 비슷해진다. 그러면 데이터 복제본이 중복되는 경우가 많아 데이터 접근성이 크게 향상되지 않는다.

DAFN (Dynamic Access Frequency and Neighborhood)은 SAF의 단점인 데이터 복제본의 중복을 해결하기 위한 방법으로 고안되었다. 먼저 SAF를 사용하여 모든 노드에 데이터 복제본을 할당한다. 이후 노드들은 자신과 연결되어 있는 이웃 노드들이 가지고 있는 데이터 복제본과 자신이 가지고 있는 데이터 복제본을 비교하여 중복되는 데이터 복제본이 있을 경우 더 낮은 접근 빈도를 가지는 노드가 다른 데이터의 복제본을 할당받아 교체함으로써 데이터 복제본의 중복을 최소화한다.

DCG (Dynamic Connectivity based Grouping)는 노드들의 연결 정보를 이용하여 그룹을 형성하여 그룹 내의 노드들끼리 접근 빈도를 비교하여 데이터를 할당받는 방법이다. 노드들의 연결 정보에 의해 이중결합요소 (bi-connected component)를 이용하여 그룹이 나누어지며, 그룹에 속해있는 노드들의 데이터의 접근 빈도를 모두 더한다. 그리고 데이터 접근 빈도 합을 정렬하여 그룹 내에서 접근 빈도가 높은 데이터부터 그 데이터 접근 빈도가 높은 노드에 데이터 복제본을 할당한다. 이런 방법을 통해 그룹에 속해있는 노드들은 데이터 중복없이 공유할 수 있으며, 앞에서 언급한 두 방법보다 높은 데이터 접근성

을 보여준다.

위 세가지 데이터 복제본 할당 기법들은 움직이는 노드의 이동성 때문에 물리적으로 연결이 끊기는 것만 고려하였다. 하지만, 본 연구에서는 형성되는 그룹 내에 이기적인 노드가 포함되는 경우도 고려하여 데이터 접근성을 향상시키고자 한다.

2. 이기적인 노드들의 성향

모바일 애드혹 네트워크에서 라우팅과 관련되어 이기적인 모바일 노드를 찾아내는 연구는 이미 활발히 진행되고 있다. 모바일 애드혹 네트워크에서는 움직이는 노드가 각각 라우터의 역할을 하고 있기 때문에 이기적인 노드를 제외하고 연결하는 네트워크 전체의 안정성을 가지려고 한다. 특히 [11]에서는 네트워크 흐름을 방해하는 이기적인 성향을 가진 노드 유형을 라우팅 프로토콜와 연관하여 아래와 같이 세 가지 유형으로 정의하였다 [12].

1) 유형1(SN1: Selfish Nodes Type1)

유형1의 노드들은 DSR [13] 라우팅 경로(Routing Path) 찾기와 경로 유지 단계에는 참여는 하지만 크기가 큰 데이터 패킷의 전달은 거절한다 (데이터 패킷은 일반적으로 라우팅 제어 패킷보다 크다).

2) 유형2 (SN2: Selfish Nodes Type2)

유형2의 노드들은 DSR 라우팅 경로를 찾는 단계에도 포함되지 않을 뿐더러 데이터 패킷 전달도 하지 않는다. 자신의 자원 (Resource)을 오로지 자신에게 이득이 되는 패킷을 전송하는 데에만 사용한다.

3) 유형3 (SN3: Selfish Nodes Type3)

유형3의 노드들은 자신의 자원에 기반하여 서로 다르게 행동을 한다. 이 노드들은 상위 임계점 T1, 하위 임계점 T2를 두어 자신의 자원 상태가 최상인 E와 T1 사이일 때에는 이기성을 가지지 않고 네트워크 안에서 적절하게 자신의 역할을 한다. 하지만 노드의 자원 상태가 T1과 T2사이일 때에는 SN1과 같이 크기가 작은 패킷은 전달하지만 큰 데이터 패킷의 전달은 하지 않는다. T2보다 작을 경우 SN2와 같이 행동한다. E, T1, T2의 관계는 E>T1>T2 이다.

위의 세 유형은 모두 이기적인 성향을 가진 노드를 나타낸다. 하지만 SN2의 경우는 라우팅 과정에서 쉽게 제거될 수 있지만, SN1, SN3는 라우팅 과정에서 찾아내기가 어려우며, 네트워크 계층을 넘어서 데이터 관리 측면에서 악영향을 미치기 쉽다.

하지만 아직까지 데이터 관리 측면에서 이기적인 모바일 노드에 관련되어 정의된 바가 없으며, 아직까지 연구된 바가 없다. 본 연구에서 이런 이기적인 모바일 노드의 성향을 데이터 관리 측면에 맞추어 새롭게 정의한다. 또한 그러한 이기적인 노드를 찾아내어 네트워크에 영향을 최소화하는 방법을 제안한다.

3. VaR (Value at Risk)

VaR[14] 모형이란 금융학에서 사용되는 개념으로 투자를 할 때 원금손실의 위험도를 확인하기 위해 사용하는 도표로써 기업의 포트폴리오를 통해 시장위험으로 인한 금융자신의 변화로 인해 생겨나는 최대 손실예상액을 추정할 수 있는 수치로 표현되는 모형이다. 즉, VaR는 위험의 정도를 적용한 기업의 가치를 나타낼 수 있으며, 정상적인 시장조건을 전제하에 시장위험으로 발생할 수 있는 최대 손실 예상액을 추정한 수치이다.

VaR를 계산하기 위해서는 보유기간과 신뢰수준을 필요로 하며, 보유기간은 상품의 종류와 포지션의 규모 등에 대한 의사결정에 의해 설정되며, 보유기간이 길수록 손실 발생 확률도 높아져 VaR값이 증가한다. 신뢰수준은 확률개념처럼 가격의 유동성이 일정한 범위에서 있을 확률을 의미하며, 신뢰수준이 높아지면 VaR값도 증가하고 손실가능성은 낮아진다.

VaR 모델은 금융학에서 사용되는 모델이기 때문에 VANET 환경에 적합하지 않다. 본 연구에서는 기존 VaR모델의 수식에 포함되는 변수에 변화를 줘서 이기적인 모바일 노드를 찾아내기 적합한 형태로 변환한다.

Ⅲ. 이기적인 노드가 가지는 특성

이 장에서는 네트워크 단계에서 라우팅 프로토콜에 맞춰진 이기적인 노드들의 성향이 아닌 데이터 관리 단계에 적용되는 이기적인 모바일 노드들의 특성을 새롭게 정의한다.

1. 이기적인 노드 특성1

첫 번째 이기적인 노드들의 특성은 자신의 자원 사용을 최소화하려고 한다. 이러한 특성을 가진 이기적인 노드는 다른 노드가 가진 데이터를 받으려고만 하고, 다른 노드들에게 자신이 가진 데이터를 주지 않으며, 자신의 저장소는 자신을 위해서만 사용한다.

데이터 접근성을 향상시키기 위해서 데이터의 복제와 공유가 원활히 이루어져야 하지만 이러한 특성을 가지는 노드가 공유 그룹에 포함되어 있다면 할당받은 데이터 복제본을 주려고 하지 않고 자신만 이용하려고 하기 때문에 데이터 접근성을 하락시킨다.

2. 이기적인 노드 특성2

두 번째 이기적인 노드들의 특성은 라우팅과 관련된 이기적인 노드 유형 중 첫 번째 유형과 같이 크기가 작은 패킷은 전달하지만 크기가 큰 데이터 패킷은 자신의 에너지 소비가 크기 때문에 전달하지 않는다. <그림 2>와 같이 이기성을 가진 노드는 패킷의 크기가 작은 데이터 요청 패킷은 전달을 해주지만 데이터를 실어 크기가 큰 패킷은 전달하지 않고 무시한다.

이 특성은 목적지가 자신이 아닌 패킷을 다른 노드에게 전달할 때 자신의 에너지가 소비되지 않고자 하기 때문에 생겨난다. 크기가 작은 패킷은 에너지 소비가 크지 않지만, 크기가 큰 데이터 패킷의 경우 에너지 소비가 크기 때문에 크기가 큰 데이터 패킷의 경우만 무시한다.

3. 이기적인 노드 특성3

네트워크에 참여하고자 하며 애드혹 네트워크 또는 애드혹 네트워크의 데이터 공유 그룹에 참여하고자 한다. 이기적인 노드 세 번째 특성은 더 많은 데이터에 접근하기 위해 데이터를 받기를 원하기 때문에 최대한 공유그룹에 참여하고자 하며, 공유그룹에 참여하지 못한다면 애드혹 네트워크라도 포함되기 위해서 노력한다. 그렇기 때문에 이러한 노드들은 자신의 거짓된 정보를 보내서라도 네트워크 또는 데이터 공유 그룹에 참여하고자 한다.

Ⅳ. 제 안 기 법

이 장에서는 차량 간의 애드혹 네트워크에서 데이터를 안정적으로 전송하는 것을 방해하는 이기적인 노드를 제거하기 위한 방법인 Friendship-VaR를 제안한다.

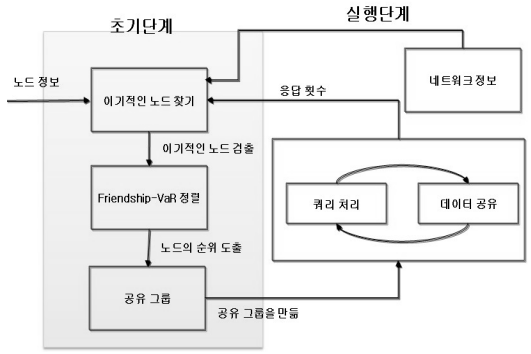
1. Friendship-VaR

Friendship-VaR (Friendship Value at Risk)는 기존 금융학에서 투자이익을 예측하기 위한 모형인 VaR (Value at Risk) 모형을 본 연구에 접목시킨 것이다.

본 연구에서 제시한 Friendship-VaR는 VaR의 개념을 이용하여 이기적인 노드들을 제거하기 위하여 제시하였다. 실제로 금융학에서도 전혀 정보가 없는 기업의 포트폴리오를 통해 신뢰성을 확보하기 위한 방법으로 사용되었듯이, Friendship-VaR는 정보를 알 수 없는 노드가 이기성을 가졌는지 가지지 않았는지를 판단하여 노드들간의 신뢰성을 확보하는데 사용된다. 기존의 금융학에서 사용하는 VaR에서는 그 값이 커질수록 위험이 커지는 것[14]과 같이 Friendship-VaR 역시 값이 커질수록 신뢰도가 떨어진다.

2. Friendship-VaR의 전체 과정

<그림 3>은 Friendship-VaR를 계산하는 전 과정을 도식화한 것이다. 이 과정은 크게 두 부분으로 나뉜다. 초기 단계(Initial Phase)와 실행 단계(Running Phase)가 있는데, 초기 단계는 음영으로 표시된 부분으로 노드의 정보들을 이용하여 Friendship-VaR 수식



<그림 3> Friendship-VaR의 전체 과정
<Fig. 3> Overall process of Friendship-VaR

에 적용한다. 노드들은 각 노드들에 대해 계산된 Friendship-VaR의 값을 비교하여 이기적인 노드라고 판단되는 것들을 제외하고 데이터 공유 그룹을 결정한다.

실행 단계에서는 실제로 노드가 자신이 필요로 하는 데이터의 원본 또는 데이터 복제본을 가지고 있는 노드들에게 요청을 한다. 요청을 받은 노드는 자신의 저장소에 있는 데이터를 요청한 노드에 보내주며 공유를 한다.

실행단계서는 초기 단계에서 결정된 Friendship-VaR를 보완하는 역할을 한다. 왜냐하면 초기단계에서 자신의 정보를 주고 받을 때 이기적인 노드는 Friendship-VaR를 계산하기 위한 알려주는 정보들을 실제 정보가 아닌 거짓된 정보를 줌으로써 속일 수가 있기 때문이다. 그렇게 Frienship-VaR를 낮춰 신뢰성을 획득할 수 있기 때문에 실행 단계에서 이러한 점을 보완하고자 한다. 자세한 보완 방법은 4.5절에서 자세하게 설명한다.

3. Friendship-VaR의 요소

기존의 VaR의 수식은 **기댓값(Expected Loss) = 보유자산(Exposure at Default) * 신뢰수준(Probability of Default) * 주가변동성(Loss Given at Default)]**이다 [14]. 기존의 VaR 수식을 VANET에 그대로 적용시킬 수 없다. 그 이유는 오랫동안 누적된 데이터로부터 사람들이 판단하기 때문이다. Friendship-VaR는 기존 VaR의 각 요소의 변경을 통해 유도된다.

보유자산 (Exposure at Default) - 보유자산은 기업이 자산을 얼마나 가지고 있는 지를 판단할 수 있게 해준다. VANET에서는 노드들이 가지고 있는 자원들을 보유자산으로 볼 수 있다.

신뢰수준 (Probability of Default) & 주가변동성 (Loss Given at Default) - 신뢰수준은 확률분포로부터 VaR가 얼마나 신뢰성을 가질 수 있는지를 알려준다. 주가 변동성은 시장조건에 의해 발생될 수 있는 변동성을 의미한다. Friendship-VaR에서는 이 두 가지 VaR의 요소를 위험(Risk)로 정의하였으며 위험정보에는 노드가 요청한 데이터에 대한 응답이 오지 않을 확률과 이기적인 노드들 때문에 전송되지 않는 데이터의 개수가 포함된다.

VANET에서는 각 노드들의 신뢰도를 측정하기 위해서 하나의 노드가 전체 노드들에 대해 Friendship-VaR를 계산한다. 신뢰도를 측정하기 위해서 사용되는 Friendship-VaR의 수식은 **기대치(Expected Reliability) = 자산(Property) * 위험(Risk)** 로 이루어진다. 기존 VaR의 수식에서 보유자산은 자산으로 치환하고, 신뢰수준과 주가변동 성 두 요소를 묶어서 위험으로 치환된다.

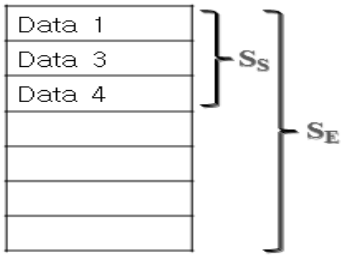
Frinedship-VaR의 기대치가 커질수록 이기적인 노드일 확률이 높은 것으로 판단되며 다음 장에서는 Frienship-VaR의 각 요소에 대해서 설명한다.

1) 자산

자산은 $P=\frac{1}{S*K}$ $(단, K\geq 1)$로 표현된다. 노드들의 자산으로 표현되는 것은 각 노드가 사용가능 한 저장소의 크기(S)와 가지고 있는 데이터의 종류 개수(K)이다. 사용가능 한 저장소 크기(S)는 노드의 전체 저장소의 크기(S_E) - 공유하는데 사용한 저장소 크기(S_S)로 나타낸다. <그림 4>를 보면 노드의 총 저장소 크기가 7이라고 했을 때, 공유를 위해 사용 중인 저장소 크기가 3이라고 하면 S=7-3으로써 4가 된다.

변환하여 P에 적용시키면

$P=\frac{1}{(S_E-S_S)*K}$ $(S_E>S_S, K\geq 1)$로 표현된다. 사용



<그림 4> 노드의 저장소
<Fig. 4> Storage of node

가능 한 저장소가 클수록 자산의 값이 작아지므로 Friendship-VaR의 크기가 작아진다.

K는 노드가 가지고 있는 데이터 종류 개수로 표현되는데 각 노드들은 자신의 원본 데이터를 가지고 있기 때문에 항상 1보다 크게 된다. <그림 4>를 보면 K는 3으로 볼 수 있다. 그렇다면 $P=\frac{1}{(7-3)*3}$가 되어 $\frac{1}{12}$가 된다. K가 커질수록 자산의 값이 작아지므로 Friendship-VaR와는 반비례관계이다.

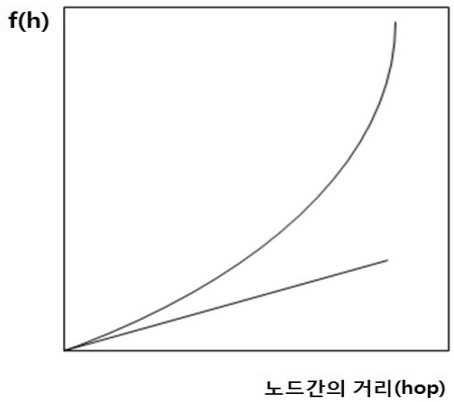
2) 위험

위험은 $R=\frac{f(h)}{CC}$로 표현된다. 모든 노드들에 대한 신뢰도를 계산 하기 위해서는 자산과 함께 위험에 대한 요소가 포함되어야 한다. 위험을 계산하기 위해서 포함되는 요소는 각 노드들과의 연결 횟수(Connectivity Counter)와 해당 노드와의 거리(f(h))이다.

연결 횟수(CC)는 연결테이블(Connectivity table)을 통해서 계산된다. 연결 횟수는 네트워크 측면에서의 연결을 의미하는데, VANET에서는 움직이는 노드들 간의 연결 횟수가 많을수록 두 노드가 가까운 거리에 있을 확률이 높고, 이동방향이 같을 수 있다고 판단할 수 있다. 연결테이블은 <표 1>과 같은 형태이

<표 1> 노드간 연결 횟수
<Table 1> Connectivity Counter between nodes

노드 ID	연결 횟수(CC)
1	5
2	10
3	1



<그림 5> 노드간의 거리
<Fig. 5> The number of hop between nodes

며 각 노드들과의 연결 횟수를 기록한다. 연결 횟수의 기본 값은 1로 두며 해당 노드와 연결이 이루어질 때마다 1씩 증가시킨다. 연결횟수가 커질수록 근접해 있다고 판단되기 때문에 $\frac{1}{CC}$ 값을 위험요소로 포함된다. f(h)는 신뢰성을 측정하고자 하는 대상 노드와의 거리(hop)을 통해서 계산된다. 대상 노드와의 거리가 멀수록 연결이 쉽게 끊겨 신뢰성이 떨어지기 쉽기 때문이다.

Friendship-VaR에서는 <그림 5>와 같이 모바일 호스트 간의 거리를 기하급수적인 함수에 적용하여 그 결과 값을 사용함으로써 거리의 영향을 많이 받도록 한다. 노드 간의 거리가 클수록 Friendship-VaR는 커지므로 비례관계가 형성된다.

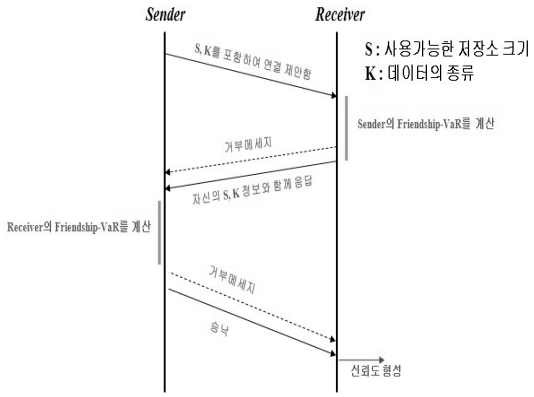
각각의 요소로 결합된 Friendship-VaR를 보면

$$Friendship-VaR = property*risk = \frac{1}{(S_E-S_S)*K}*\frac{f(h)}{CC} = \frac{f(h)}{(S_E-S_S)*K*CC} \quad (\text{단}, S_E > S_S, K \geq 1, CC \geq 1)$$

와 같이 표현된다. 이렇게 표현된 Friendship-VaR의 값이 클수록 이기적인 노드일 확률이 높다.

4. Friendship-VaR의 프로토콜

<그림 6>은 Friendship-VaR를 구하기 위한 두 노드



<그림 6> Friendship-VaR의 프로토콜
<Fig. 6> Friendship-VaR ‘s protocol

간의 프로토콜을 나타낸 것이다. 먼저 이기성을 측정하고자 하는 노드(Sender)가 대상 노드(Receiver)에게 자신의 사용가능한 저장소 크기(S)와 현재 가지고 있는 데이터의 개수(K)를 보내주면서 신뢰관계를 맺고자 요청을 한다. 상대 모바일 호스트(Sender)의 정보를 받은 모바일 호스트(Receiver)는 보내준 정보를 바탕으로 Friendship-VaR를 계산한다. 계산 후 이기성을 갖고 있다고 판단이 되면 거부 메시지(Denial Message)를 보내 종료한다. 반면에 이기성을 갖고 있지 않다고 판단되면 자신의 사용가능한 저장소 크기(S)와 현재 갖고 있는 데이터의 개수(K)를 보내주고, 그 정보를 받은 노드(Sender)는 Friendship-VaR를 계산하여 이기적인 노드인지 판단한다. 여기서 최종적으로 이기성을 갖고 있다고 판단되면 거부 메시지를 보내 종료하지만, 이기성을 갖고 있지 않다고 판단이 되면 서로의 신뢰관계가 형성이 되어 서로 데이터를 공유하고자 한다.

5. Friendship-VaR의 보완

4.2절에서 언급한 초기 단계에서의 문제점을 보완하기 위해 본 연구에서는 <그림 3>에서 오른편에 있는 실행 단계에서 다른 노드에게 데이터를 요청하였을 때 오는 응답을 통해 이기적인 노드인지 아닌지 판단을 할 수 있는 요소를 추가하였다. 추가한 요소는 데이터 요청에 대한 응답횟수(K_C)인데, 네트워크

<표 2> 응답 횟수
<Table 2> The number of reply

노드 ID	KC
1	3
2	10
3	0

에 연결이 되어있으며, 해당 데이터를 가지고 있는 노드가 요청하는 데이터를 전송해주면 1을 추가하고, 응답이 없으면 1을 감소시킨다.

$$K_C = \begin{cases} K_C + 1 & \text{if } reply \\ K_C - 1 & \text{if } non-reply \end{cases}$$

각 노드는 K_C에 대하여 <표 2>와 같이 모든 노드들에 대한 테이블을 가지고 실행 단계에서 응답의 결과를 통해 테이블의 값을 변화시킨다.

위와 같은 형식으로 K_C를 삽입하여 기존의 초기 단계에서 발생할 수 있는 이기적인 노드들의 속임수를 방지할 수 있다. 실행 단계에서는 계속적으로 K_C를 측정하여 Friendship-VaR가 임계점(Threshold)이하로 떨어져 이기적인 노드라고 판단할 수 있다. 보완된 Friendship-VaR의 수식은 아래와 같이 되며 값이 커지면 커질수록 이기성이 가지는 노드로 판단한다.

$$Friendship-VaR = a^* \frac{f(h)}{(S_E - S_S)^* K^* CC} + (1-a)\frac{1}{K_C}$$
$$(\text{단}, S_E > S_S, K \geq 1, CC \geq 1, 0 < a < 1)$$

Ⅴ. 실험결과

이 절에서는 본 연구에서 제안한 기법에 대한 성능을 평가하기 위해서 C언어 기반의 CSIM[15]을 이용하였다. 또한 시뮬레이션을 위해 50m*50m의 크기의 이차원 공간에서 40개의 노드를 랜덤하게 배치하였으며, 이동 패턴은 자유롭게 하였다. 노드 $M=\{M_1,...,M_{40}\}$으로, 데이터는 $D=\{D_1,...,D_{40}\}$로 표현한다. 각 모바일 호스트 i는 각 데이터 i를 원본 데이터로 가진다. 또한 모바일 호스트는 최대 C개의 데이터 가질 수 있으며, 노드의 움직임에 대한 제한은 두

<표 3> 시뮬레이션 변수
<Table 3> Parameters of Simulation

변수	값
d(이동속도)	0 ~ 1(m/s)
R(통신 거리)	7(m)
C(저장소 크기)	10
T(시간 간격)	1000(s)

지 않고 자유롭게 움직이도록 하였다. 노드들의 다른 변수들은 <표 3>과 같이 고정시켰다. <표 3>에서 변수를 살펴보면 d는 노드의 이동 속도를 나타내며 0 ~ 1m/s로 랜덤하게 설정하였다. T는 Friendship-VaR의 초기 단계를 계산하거나 각 노드에 데이터를 할당하는 기간을 말한다. 즉, Friendship-VaR를 계산 후 다음 계산을 할 때까지 걸리는 시간으로, 1000초로 설정하였다. R은 노드의 통신 범위로서 본 연구에서는 노드의 통신 범위(R)을 반경 7m로 설정하였다. 그리고 C는 모바일 호스트가 가질 수 있는 데이터 개수를 뜻한다. C는 10으로 설정하였으며, 실제로 실험을 통해 살펴본 결과 R과 C의 변화는 실험의 영향에서 같은 결과를 보여주었기 때문에 실험에 크게 영향을 미치지 못한다고 판단하였다.

실험은 기존의 데이터 복제본 할당 방식에서 가장 성능이 좋은 DCG[8]를 바탕으로 이기성을 제거하는 Friendship-VaR를 적용하였으며, Friendship-VaR의 초기 단계와 실행 단계 계산을 위한 가중치(a)값은 0.6으로 두었다.

본 연구에서는 저장소 크기를 고려하여 이기적인 모바일 호스트를 제거하는 기법을 처음으로 제안하였기 때문에 네트워크 단계에서의 이기적 노드를 제거하는 기법과는 비교할 수 없다. 그렇기 때문에 본 실험에서는 현재 데이터 복제본 할당 기법 중 가장 좋은 성능을 보이는 DCG와 쿼리 지연과 데이터 접근성을 비교하여 성능을 평가하고자 한다.

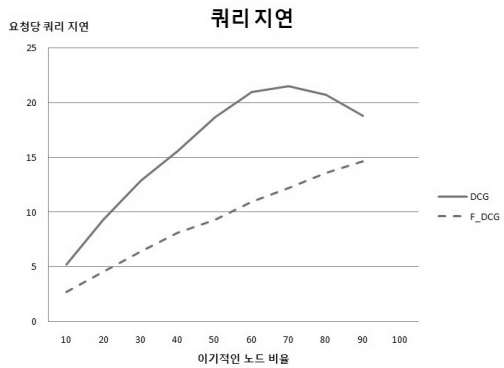
쿼리 지연은 노드가 데이터를 요청하였을 때 응답이 올 때까지 걸리는 시간을 측정하였으며, 데이터 접근성은 노드가 데이터를 요청했을 때 응답이 올 가능성을 측정했다. 아래의 실험을 통해 쿼리 지연이 줄어들어 네트워크 효율이 증가하고 데이터 접근성이 증가함을 알 수 있다.

또한 모든 경우에 대해서 50,000단위 시간에 대하여 측정하여, 쿼리 지연과 데이터 접근성의 평균값을 취하였다.

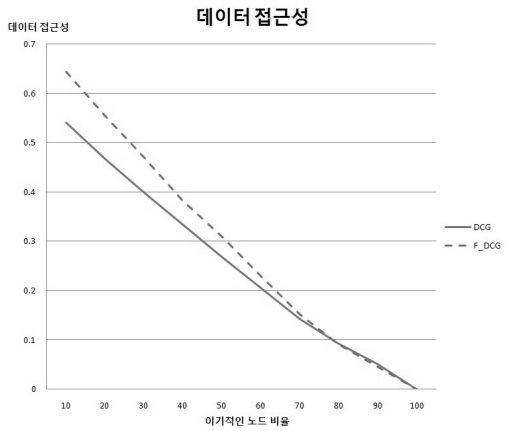
1. 쿼리 지연

먼저 쿼리 지연에 대해서 살펴보면, 이기적인 노드의 수가 많을수록 쿼리에 대한 지연이 늘어난다. 서론에서 언급한 바와 같이 이기적인 노드들은 크기가 작은 데이터 요청 패킷은 전달을 해주지만 크기가 큰 데이터 패킷은 전달을 하지 않는다. 그렇게 되면 데이터를 요청한 노드는 요청에 다한 응답 메시지를 받지 못하고 요청한 데이터를 계속 기다리게 된다. 노드는 요청 패킷의 생명 주기(Life Time)가 끝날 때까지 기다리게 되며, 쿼리 처리는 지연이 된다.

<그림 7>의 그래프를 보면 Friendship-VaR를 적용하지 않은 DCG[8]는 일정 수준까지 쿼리 지연이 증가하다가 이기적인 노드의 비율이 70%가 되면 쿼리 지연이 낮아진다. 그 이유는 이기적인 노드가 많아지면 쿼리의 지연이 일어나기 거부 메시지를 보내기 때문에 쿼리의 지연이 작아진다. 하지만 Friendship-VaR에서는 데이터를 요청할 때 이기적인 노드들을 통하지 않고 자신이 신뢰를 할 수 있는 노드들로부터 데이터를 가지고 오기 때문에 쿼리 지연이 낮아지며, 이기적인 노드가 많으면 많을수록 쿼리의 지연이 증가하게 된다. 이를 통해 Friendship-VaR를 이



<그림 7> 요청 당 쿼리 지연
<Fig. 7> Query delay per request



<그림 8> 데이터 접근성
<Fig. 8> Data accessibility

용하게 된다면 이기적인 노드들에 의한 쿼리 지연이 감소하기 때문에 효율이 증가하게 된다.

2. 데이터 접근성

노드들의 각 데이터에 대한 접근성을 살펴보면 <그림 8>과 같이 기존의 DCG 보다 약 10%이상 향상됨을 보여준다. 기존의 DCG의 경우는 기본 조건이 안정화된 그룹을 만들어 네트워크 연결이 끊어질 확률을 최소화한다. 하지만 그룹에 이기적인 노드가 포함되어 있다면, 그룹의 안정성을 사라지게 된다. 이기적인 노드들은 자신이 데이터 복제본을 할당 받았음에도 불구하고 그룹 내의 다른 노드들에게 데이터를 전송하지 않기 때문에 데이터의 접근성을 낮아질 수밖에 없다. 하지만 Friendship-VaR를 통해 이기적인 노드들을 제거한 후 데이터 공유 그룹을 형성한다면 공유 그룹이 안정하게 형성되며 공유 그룹 내의 노드들은 공유 그룹 내의 데이터를 모두 전송 받을 수 있기 때문에 데이터 접근성이 향상되게 된다.

Ⅵ. 결론 및 향후 연구

본 연구에서는 VANET 안에서 이기적인 노드가 존재할 수 있는 가능성을 제시하고, 제거할 방법을 제안하였다. 제안한 기법은 먼저 각 노드들의 정보인 저장소의 크기와 현재 가지고 있는 데이터의 개수를 이용해 초기 단계에서 계산하며, 계산의 과정을 Friendship-VaR 프로토콜에 맞춰서 이루어진다. 실행 단계에서는 응답횟수를 통해서 초기 단계에서 다른 노드를 속인 이기적인 노드들을 제거하도록 한다. 실험 결과를 통해 데이터 접근성 측면은 약 10% 가량 증가하였으며, 쿼리 지연 또한 확연하게 감소하여 애드혹 네트워크의 효율이 증가하였다.

향후 연구로는 Friendship-VaR를 통해 적절히 이기적인 노드를 제거하고 제거되더라도 만들어진 그룹에 영향을 미치지 않는 그룹화를 하는 방법 개발을 향후 연구로 한다.

참 고 문 헌

[1] 최병철, 한승완, 정병호, 김정녀, “지능형 차량 보안 기술 동향,” *ETRI, 전자통신동향분석*, 제22권, 제1호, pp.114-118, 2007. 2.

[2] TTA, “Standardization Roadmap for IT839 Strategy-Telematics/ITS,” 2006.

[3] V. Cherfaoui, T. Denoeux, and Z. L. Cherfi, “Distributed data fusion: application to confidence management in vehicular networks,” *Proc. Int. Con. Information Fusion*, pp.1-8, July 2008.

[4] W. Chen, R. K. Guha, T. J. Kwon, J. Lee, and I. Y. Hsu, “A survey and challenges in routing and data dissemination in vehicular ad-hoc networks,” *Proc. IEEE Int. Conf. Vehicular Electronics and Safety (ICVES)*, pp. 328-333, Sept. 2008.

[5] 이상선, “VANET(Vehicle Ad-hoc Network) 환경에서의 라우팅 기술 및 서비스 개발 동향,” *KIISE*, vol. 22, no.1, 2008. 5.

[6] M. Tamori, S. Ishihara, T. Watanabe, and T. Mizuno, “A replica distribution method with consideration of the positions of mobile hosts on wireless ad hoc networks,” *Proc. Int. Conf. Distributed Computing Workshop*, pp. 331-335, July 2001.

[7] P. Bellavista, A. Corradi, and E. Magistretti

"REDMAN: A decentralized middleware solution for cooperative replication in dense MANETs," *Proc. Int. Conf. Pervasive Computing and Communications Workshops*, pp. 158-162, Mar. 2005.

[8] T. Hara "Effective replica allocation in ad hoc networks for improving data accessibility," *Proc. IEEE INFOCOM*, pp.1568-1576, April 2001.

[9] E. Adar and B. Huberman, "Free riding on gnutella," *First Monday*, vol. 5, no. 10, pp. 1-22, Oct. 2000.

[10] K. Balakrishnan, J. Deng, and P. K. Varshney, "TWOACK: Preventing selfishness in mobile Ad Hoc networks," *Proc. IEEE Wireless Communications and Networking Conf.,* pp. 2137-2142, Mar. 2005.

[11] Y. Yoo and D. P. Agrawal, "Why does it pay to be selfish in a MANET?" *IEEE Wireless Communications*, vol. 13, no. 6, pp. 87-97, Dec. 2006.

[12] P. Michiardi and R. Molva, "Simulation-based analysis of security exposures in mobile Ad Hoc networks," *Proc. European Wireless Conf.,* Feb. 2002.

[13] D. Johnson, D. Maltz and Y. Hu, "The dynamic source routing protocol (DSR) for mobile ad hoc networks for IPv4," *Internet Request for Comments RFC 4728*, Feb. 2007. http://www.ietf.org/rfc/rfc4728. txt.

[14] Wikipedia "Value at Risk," URL : http://en.wikipedia.org/wiki/Value_at_risk

[15] H. Schwetman, "CSIM19: CSIM19: a powerful tool for building system models," *Conf. Winter Simulation, Software/Modelware Ttutorials,* pp. 250-255, Dec. 2001.

저자소개

심 규 선 (Shim, Kyu-Sun)

2003년 3월 ~ 현재 : 고려대학교 컴퓨터학과 학부과정

이 명 수 (Lee, Myong-Soo)

2007년 3월 ~ 현재 : 고려대학교 컴퓨터학과 박사과정
2004년 3월 ~ 2006년 2월 : 고려대학교 컴퓨터학과 석사
1997년 3월 ~ 2004년 2월 : 고려대학교 컴퓨터학과 학사

이 상 근 (Lee, SangKeun)

2003년 ~ 현재 : 고려대학교 컴퓨터학과 부교수
2000년 ~ 2001년 : University of Tokyo 특별 방문 연구원
1996년 3월 ~ 1999년 2월 : 고려대학교 컴퓨터학과 박사
1994년 3월 ~ 1996년 2월 : 고려대학교 컴퓨터학과 석사
1990년 3월 ~ 1994년 2월 : 고려대학교 컴퓨터학과 학사