CityMob: a mobility model pattern generator for VANETs

Francisco J. Martinez†, Juan-Carlos Cano‡, Carlos T. Calafate‡, Pietro Manzoni‡

†Department of Computer and Systems Engineering
University of Zaragoza, Spain
Email: f.martinez@unizar.es

‡Department of Computer Engineering
Polytechnic University of Valencia, Spain
Email: {jucano, calafate, pmanzoni}@disca.upv.es

Abstract— **Ad hoc networking is regarded as an adequate solution to cooperative driving between communicating cars on the road. Deploying and testing these networks, usually known as *Vehicular Ad-hoc Networks* (VANETs), involves a high cost in the real world, and so simulation is an useful alternative in research. One of the most critical issues in a simulation study of VANETs is the use of a mobility model which resembles, as closely as possible, the real behavior of vehicular traffic. Mobility models are crucial to obtain accurate and meaningful simulation results.**

In this paper we present *CityMob*, a mobility pattern generator for VANETs. Citymob allows researchers to easily create urban mobility scenarios, including the possibility to model car accidents. We designed and developed it targeting compatibility with the ns-2 simulator, and we implemented three different mobility models: *Simple Model* (SM), *Manhattan Model* (MM) and *Downtown Model* (DM). Based on a flooding alert protocol we show that the most realistic mobility model to simulate traffic accidents is the Downtown model. We also find that, for flooding to be effective, a moderate number of vehicles is required.

Index Terms— **Vehicular ad hoc networks, mobility models, inter-vehicle communication.**

I. Introduction

Mobile ad hoc networks (MANETs) are a type of wireless networks that do not require any fixed infrastructure [1]. These networks are being adopted to solve situations where communication is required, but where deploying a fixed infrastructure is impossible.

More recently, ad hoc networking is regarded as the most adequate solution to cooperative driving between communicating cars on the road. Such networks, named *Vehicular Ad-hoc Networks* (VANETs), represent a rapidly emerging research field, being a particularly challenging class of Mobile Ad Hoc Networks.

VANETs have particular features like distributed processing and organized networking, large number of nodes, high node speed, constrained but highly variable network topology, signal transmissions blocked by buildings, frequent partition due to the high mobility, and, on the contrary to MANETs, no significant power constrains.

Such features make standard networking protocols inefficient or unusable in VANETs; hence, there is a growing effort in the development of specific communication protocols and methodologies for vehicular networks [2]. There are also strong economical interests since vehicle-to-vehicle communication allows to share the wireless channel for mobile applications, to improve route planning, to control traffic congestion, or to improve traffic safety, e.g., avoiding crash situations, thus saving lives on roads, or warning drivers about traffics jams [3].

Similarly to other disciplines, simulation has emerged as the best method in order to test and evaluate different VANETs solutions due to infrastructure difficulties, economic issues and other limitations. One of the most critical issues in a simulation study of VANETs is the use of a mobility model which reflects, as closely as possible, the real behavior of vehicular traffic. When dealing with vehicular mobility modeling we can distinguish between macro-mobility and micro-mobility descriptions [1]. The better we model both macro-mobility and micro-mobility descriptions, the more accurate results will be obtained.

In this paper we present *CityMob*, a mobility pattern generator that allows researchers to easily create urban mobility scenarios including the possibility to model car accidents and to use a flooding based alert protocol to announce events. We designed and developed it targeting compatibility with the ns-2 [4] simulator, and we implemented three different mobility models: *Simple Model* (SM), *Manhattan Model* (MM) and *Downtown Model* (DM).

This paper is organized as follows: Section II describes related work dedicated to the analysis of the mobility models proposed for VANETs. Section III describes the mobility models and the software tool we have developed. Section IV presents the performance evaluation of these three mobility models. Finally, Section V presents some concluding remarks.

II. Mobility models for VANETs

One of the most important issues to take into account when creating a simulation environment in VANETs is to correctly model how vehicles move. Based on a study of mobility behavior of mobile users [5], existing models try to closely represent the movement patterns of users. These models, along with appropriate VANETs scenarios, provide a suitable environment for the simulation and evaluation of ad hoc communication performance. The general problem of modeling the behavior of nodes belonging to a mobile network does not have a unique or straightforward solution. Mobility patterns are dependent on various factors, such as the physical environment, the user's objectives, and the user's inter-dependencies. Previous works [6], [7] showed that these models can greatly affect simulation results, so realistic movement patterns are compulsory when simulating VANETs.

A wide variety of mobility models have been proposed for VANET simulations. Saha and Johnson [8] modeled vehicular traffic with a random mobility of nodes over real road topologies extracted from the maps of the US Census Bureau TIGER database. In that work, nodes select one point over the graph as their destination and compute the shortest path to get there. The edges sequence is obtained weighting the cost of traveling on each road on its speed limit and the traffic congestion.

Huang et al. [9] studied taxi behavior. They model the city as a Manhattan style grid with a uniform block size across the simulation area. All streets are assumed to be two-way, with one lane in each direction. Taxi movements are constrained by these lanes. A taxi is characterized by a preferred speed, a maximum acceleration and deceleration, a speed variation associated with the preferred speed at steady state, and a list of preferred destinations, i.e., the taxi stands. The taxis are randomly assigned one of three preferred speeds.

Choffnes et al. [10] designed a street mobility model, named STRAW that incorporates a simple car-following model with traffic control to introduce vehicular congestion, which models real traffic conditions. STRAW relies on street plans to build a road map for the specified target region. It also provides at least one lane in each direction on which vehicles can move. To determine the initial positions of vehicles on the field, it uses a random street placement model that places a vehicle in a lane of a street just before an intersection. If another vehicle is already in that lane, the new vehicle is placed behind the existing one.

Haerri et al. [11] proposed a vehicular mobility simulator for VANETs, called VanetMobiSim, which employs the Intelligent Driver Model (IDM) to determine the speed of vehicles.

In Mahajan et al. [12] three different models were presented: *Stop Sign Model* (SSM), *Probabilistic Traffic Sign Model* (PTSM) and *Traffic Light Model* (TLM). The main difference of these models is basically the algorithm used to reproduce stop signs. All roads are modeled as bidirectional roads, the SSM and PTSM assume a single lane in each direction of every road, whereas TLM provides the option for modeling multiple lanes.

In this work we developed a tool that allowed us to also create realistic VANETs but being them completely compatible with the ns-2 simulation tool. This tool was developed in C, and its name is *CityMob*.

III. Description of the CityMob mobility generator

CityMob[1] is a mobility pattern generator especially designed to investigate different mobility models in VANETs, and their impact on inter-vehicle communication performance. CityMob creates urban mobility scenarios and simulates damaged cars using the network to send information to other vehicles, trying to prevent accidents or traffic jams.

We propose three different mobility models that combine a certain level of randomness, while trying to represent some realistic environments. The models are:

1) The *Simple Model* (SM): Models vertical and horizontal mobility patterns without direction changes. Semaphores are not supported either.
2) The *Manhattan Model* (MM): We model the city as a Manhattan style grid, with a uniform block size across the simulation area. All streets are two-way, with one lane in each direction. Car movements are constrained by these lanes. The direction of each node in every moment will be random, and it can not be repeated in two consecutive movements. Moreover, this model simulates semaphores at random positions (not only in crossing), and with different delays. When a vehicle meets a semaphore, it will remain stopped until the semaphore turn to green.
3) The *Downtown Model* (DM):
   This model adds traffic density to the Manhattan model. In a real town, traffic is not uniformly distributed; there are zones with a higher vehicle density. These zones are usually in the downtown, and vehicles must move more slowly than in the outskirts. In our experiments, vehicles crossing the downtown have a random speed between 25 and 60 km/h, while the speed in the outskirts, usually higher, can be selected by the user.
   The Downtown area is defined by the coordinates ($start\_x$, end_x, $start\_y$, end_y) and can never cover more than 90% of the total map area.
   Parameter p is used to establish the probability of a node being initially located inside the downtown area, and also the probability that nodes on the outskirts move into the downtown. Remember that, once a node enters this area, it will move slower. The remaining features are the same as for the Manhattan mobility model.
   Figure 1 shows an example of this model. Notice that the darker buildings area represents the downtown. Dark rectangles represent vehicles, shadowed rectangles represent vehicles stopped at semaphores, and crosses represent damaged cars sending warning packets.

We do not distinguish among different types of vehicles (cars, trucks or taxis), but only between normal vehicles and damaged ones.

The user has to set the mobility model, the total number of nodes, the simulation time, the map size, the maximum speed value, the distance between consecutive streets and the number of damaged nodes.

A. CityMob's features

The standard scenario commonly used for MANETs tend to be either not useful in VANETs or too limited in scope [9].

In our work, a city is an area sized 1 km per 1 km. Streets will be arranged in a Manhattan style grid, with a uniform block size across the simulation area (this size can be set by user). All streets are two-way, with lanes in both directions. Car movements are constrained by these lanes. Nodes will move with a random speed lower than the maximum one defined by the user, and the movement pattern will be constrained by the mobility model selected. Damaged vehicles will remain stopped during the entire simulation time.

Within our simulation framework, we generate mobility traces for the three mobility models proposed according to the following guidelines:

1) The city is simulated in the same way for the three models, with a Manhattan grid map. Map width and height

[1]`CityMob's` source code is available at http://www.grc.upv.es/

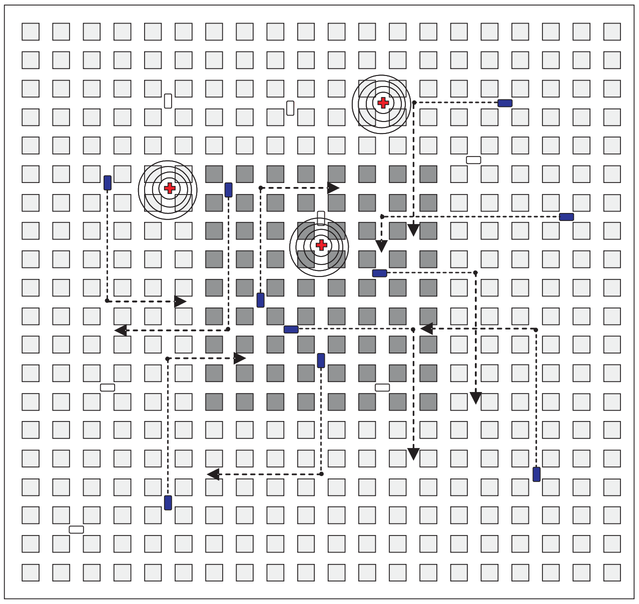

Fig. 1. Example of Downtown scenario.

are configurable parameters.

2) The distance between streets is also configurable, logically limited by map size. There must be a minimum number of crossing to allow nodes to change their direction.
3) Users can change the number of simulated nodes and the number of damaged nodes (statically located throughout the simulation area). Every node will start at a random position inside the map, although in the Downtown model the probability of starting inside the downtown is greater.
4) Speed can vary according to the map area, changing throughout the simulation. Every node will travel with a random speed for each movement, always lower than the maximum speed defined by the user.
5) There will be no accidents or crashes between nodes (in the same or different direction). We assumed that the number of lanes are enough for the vehicles that are traveling in the city.

We have to mention the presence of semaphores (for the Manhattan and Downtown models). Nodes will simulate semaphores by stopping themselves randomly. This way, simulation is more realistic since in a city semaphores are not systematically distributed on streets, and it also help us to model other unforeseen traffic events, e.g., a vehicle suddenly stops. This characteristic is one of the basic differences with the work in [9].

Figure 1 shows an example of an urban map generated by CityMob. In this case, the city is defined with a size of 1000x1000 meters, with a distance between streets of 50 meters and with a downtown area.

B. Installation and operation

CityMob has been implemented using the C programming language and it is distributed under a GNU/GPL license.

The use of CityMob is very simple. We only have to execute the application followed by the required parameters in order to generate the desired trace file.

An example of CityMob could be:

```
./citymob -m 3 -n 25 -t 1200 -s 40 -w
1000 -h 1000 -d 50 -a 3 -x 300 -y 300 -X
700 -Y 700 -p 0.80
```

In current version of CityMob, an accident is modeled keeping a node static along the simulation. The position of these nodes will be random.

IV. Performance evaluation

This section reports the results of the analysis we have performed, adopting the three mobility models described in Section III.

As usual with MANET simulations, the number of parameters and their possible values is very large. We therefore performed a thorough evaluation.

Each simulation run lasted for 60 seconds in a 1000 x 1000 meters (1 km^2) scenario, with a uniform block size of 50 x 50 meters and the number of nodes is 25, being 3 of them damaged. The maximum speed of vehicles is of 40 m/s, and in the Downtown model, downtown is established with the following coordinates: $start\_x = 300m$, $end\_x = 700m$, $start\_y = 300m$, $end\_y = 750m$; the probability of starting inside the downtown is 0.70.

Each damaged vehicle periodically broadcasts information about itself. So, the routing protocol used was *flooding*. When a vehicle receives a broadcast message, it stores and immediately forwards it by re-broadcasting the message.

Notice that in our scenarios, warning messages should be propagated to all neighbors up to a certain number of hops. Hence, the use of flooding fits our purpose adequately.

In some environments, flooding can generate many redundant transmissions and saturate the network, which may cause the well-known broadcast storm problem where serious redundancy, contention and collisions produce information losses [13].

Nevertheless, for our requirements, this protocol is very useful because we desire that the warning packets sent by damaged nodes can be received by all vehicles of the map, and this protocol offers the best reliability in terms of coverage.

Figure 2 shows an example of flooding transmission started from a damaged node (the one with the cross on top).

In simulations we use three damaged nodes, and the results we extract from our experiments are: percentage of nodes which receive traffic events, the instant of reception and their position when they do it. If a node has already received the packet, the information is discarded. Also, it is possible that some nodes never receive the packet, whether they are too isolated or because of packet losses, (e.g., due to a collision). With all these values, we can calculate some interesting data, such as the distance from source node to the destination one, the total time for a packet to arrive to all nodes, or, sometimes the number of blind nodes, i.e. nodes which never received the packet.

We now proceed to analyze the simulations results.

The Simple model only generates horizontal and vertical movements without changes of direction, and so it is easy that

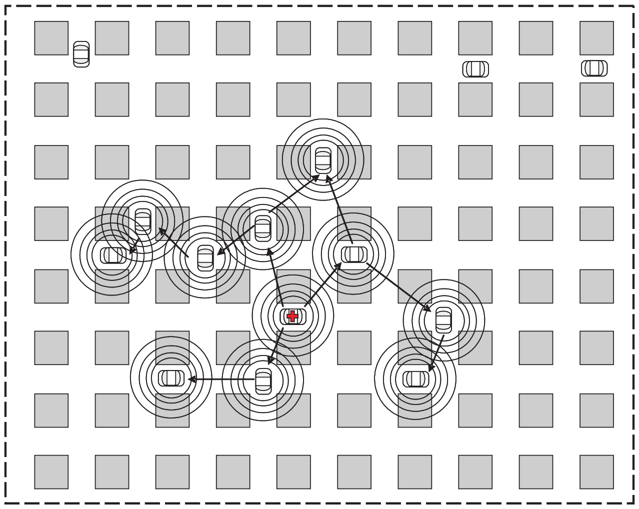

Fig. 2. Example of flooding transmission from a damaged node.

two separated nodes are not close enough to establish communication. So, in this model, the probability that an isolated node could not receive the packet sent by a damaged node is bigger than for the rest of the models.

Figure 3 shows the propagation of the packets along the nodes of the map in Manhattan Model.

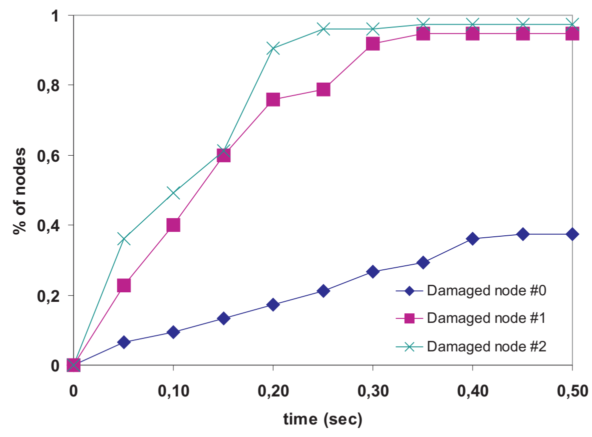

Fig. 3. Propagation of warning packets from the 3 damaged nodes along the time (Manhattan Model).

In the Manhattan model the simulation of a urban scenario is a bit more realistic: cars can take whatever direction when arriving to a crossing. So, in contrast to the simple model, each node can travel around the whole area.

Although this model is more elaborated than the previous one, the obtained results show that, in the selected scenarios, the behavior is very similar to the Simple Model.

Figure 4 shows the obtained results under the Downtown model. In this case, nodes closer to damaged vehicles are informed about the position of the damaged vehicles in a short time (quicker than for the Simple and Manhattan models), while remote nodes were not always informed of this problem.

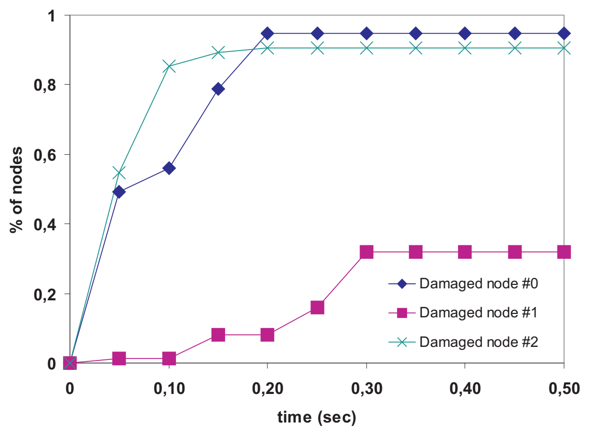

Fig. 4. Propagation of warning packets from the 3 damaged nodes along the time (Downtown Model).

The Downtown model simulates cities with beltways and more traffic in the downtown. The results obtained show that message dissemination is the most efficient of the three models. In our opinion, it is very important that vehicles are informed about damaged nodes stopped on the road very quickly, especially when they are close to them. If nodes are far, it would be better if this information is never delivered.

When damaged nodes are in a certain area of the city they are not affected by traffic in the downtown. This is demonstrated by the high number of nodes which never received packets sent by isolated nodes (see packets sent by node 1 in Table I)

This demonstrates the similarity between this model and real traffic. When we have a damaged vehicle in the outskirts, the probability of a packet arriving to downtown is lower, improving system performance and reducing unnecessary network traffic. Besides, we checked that the system works correctly when there are a lot of vehicles, and the time required to distribute packets in the downtown is lower (about half time in most cases) than the time necessary for the outskirts.

TABLE I

SUMMARY OF BLIND NODES IN DOWNTOWN MODEL.

Event	Number of blind nodes
First packet node 0	2
Second packet node 0	1
Third packet node 0	1
First packet node 1	1
Second packet node 1	24
Third packet node 1	24
First packet node 2	2
Second packet node 2	3
Third packet node 2	2

Figure 5 shows a comparison of the three models. The information dissemination latency is lower for the Downtown Model because the number of vehicles informed increases rapidly. As referred earlier, Simple and Manhattan models seem very similar. Notice that the Downtown model achieves a lower ratio in term of flooding effectiveness. Such behavior is expected since the vehicular node density is not uniform throughout the simulation area. This effect could be beneficial since isolated nodes are not informed of distant vehicles.

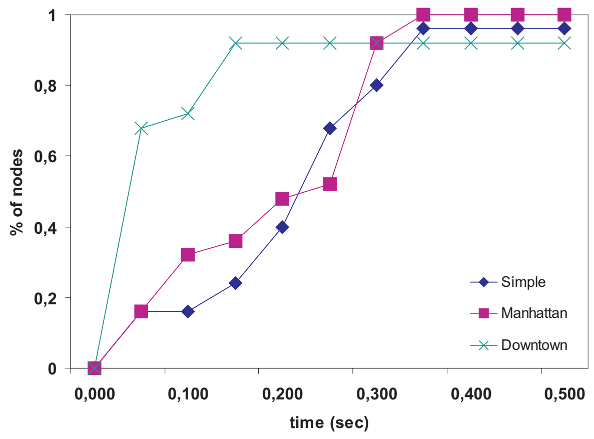

Fig. 5. Comparison of information dissemination latency in the three models.

V. Conclusions

In this paper we made a performance analysis of inter-vehicle communication systems when realistic mobility is considered.

We presented and described a mobility pattern generator called *CityMob* to be used with the ns-2 simulator. We implemented three different mobility models: the *Simple Model* (SM), the *Manhattan Model* (MM) and the *Downtown Model* (DM). We then performed a sensibility analysis to obtain more detailed results by comparing the behavior of the system with each mobility model. The results show how deeply the level of similarity with respect to real traffic can affect the performance results.

Both Simple and Manhattan models showed a similar behavior, although packet propagation is slightly quicker for the latter. In any case, the Downtown model shows better results since its level of realism is also reflected in term of flooding behavior, allowing information to be disseminated more quickly among nodes in the city. This effect is mainly due to the degree of proximity between vehicles, allowing the signal to propagate more efficiently.

It is expected that as increases, there will be too many collisions, causing packet losses. In such a case, it would be necessary a more detailed study in order to asses what is the maximum number of nodes that an area can admit without performance lost.

Finally, the flooding algorithm does not predict situations as when a vehicle receives a message from a damaged node being far away from it. This signal could travel through the whole city if there are enough nodes in the path. This, can be beneficial if the receiver's destination is near a damaged node, but in most cases this information is irrelevant and produces unnecessary network traffic. Hence, an adequate propagation limit must be sought.

We think that much work can be done in this field, e.g. testing different routing protocols, improving flooding mechanisms, finding the maximum number of nodes admitted, reducing collisions and unnecessary traffic, increasing the level of realism of the modeled system, finding new metrics, varying environments and studying systems deeply.

As future work we plan to address all these issues under an improved version of the Downtown model and to evaluate the impact that the number of nodes and traffic have over performance metrics.

Acknowledgments

This work was partially supported by the *Ministerio de Educacin y Ciencia*, Spain, under Grant TIN2005-07705-C02-01.

References

[1] M. Fiore, J. Haerri, F. Filali, and C. Bonnet, "Vehicular mobility simulation for vanets," in *Proceedings of the 40th Annual Simulation Symposium (ANSS 2007), Norfolk, Virginia*, March 2007.

[2] L. Wischhof, A. Ebner, and H. Rohling, "Information dissemination in self-organizing intervehicle networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 1, pp. 90–101, March 2005.

[3] M. Bechler, W.J. Franz, and L. Wolf, "Mobile internet access in fleetnet," in *Verteilten Systemen KiVS 2003*, February 2003.

[4] K. Fall and K. Varadhan, "ns notes and documents." The VINT Project. UC Berkeley, LBL, USC/ISI, and Xerox PARC, February 2000, available at http://www.isi.edu/nsnam/ns/ns-documentation.html.

[5] C.-K. Toh, *Ad Hoc Mobile Wireless Networks: Protocols and Systems*. Prentice Hall, 2001.

[6] Xiaoyan Hong, Taek Jin Kwon, Mario Gerla, Daniel Lihui Gu, and Guangyu Pei, "A mobility framework for ad hoc wireless networks," *Proceedings of the Second International Conference, MDM 2001 Hong Kong, China, LCNS Vol. 1987*, pp. 185–196, January 2001.

[7] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, no. 5, pp. 483–502, 2002.

[8] A.K. Saha and D.B. Johnson, "Modeling mobility for vehicular ad hoc networks," in *ACM Workshop on Vehicular Ad Hoc Networks (VANET 2004), Philadelphia PA*, October 2004.

[9] E. Huang, W. Hu, J. Crowcroft, and I. Wassell, "Towards commercial mobile ad hoc network applications: A radio dispatch system," in *Sixth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2005), Urbana-Champaign, Illinois*, May 2005.

[10] D.R. Choffnes and F.E. Bustamante, "An integrated mobility and traffic model for vehicular wireless networks," in *ACM Workshop on Vehicular Ad Hoc Networks (VANET 2005), Cologne, Germany*, September 2005.

[11] J. Haerri, M. Fiore, F. Filali, and C. Bonnet, "Vanetmobisim: generating realistic mobility patterns for vanets," in *ACM Workshop on Vehicular Ad Hoc Networks (VANET 2006), Los Angeles, California*, September 2006.

[12] A. Mahajan, N. Potnis, K. Gopalan, and A. Wang, "Evaluation of mobility models for vehicular ad-hoc network simulations," in *IEEE International Workshop on Next Generation Wireless Networks (WoNGeN 2006), Bangalore, India*, December 2006.

[13] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom 1999), Seattle Washington*, August 1999.