UrbiSim: A Framework for Simulation of Ad Hoc Networks in Realistic Urban Environment

Guillaume-Jean Herbiet
Université du Luxembourg
FSTC - CSC
Luxembourg
guillaume.herbiet@uni.lu

Pascal Bouvry
Université du Luxembourg
FSTC - CSC
Luxembourg
pascal.bouvry@uni.lu

***Abstract*—In this paper, we present UrbiSim, a new simulation framework for delay tolerant and ad hoc network research applied to urban environments. By providing two levels of mobility modeling, this work tries to cope with two of the current limitations of simulation models: realistically mimicking mobility at an individual level and offering a global movement pattern, reflecting the social behavior of users. Conceived as a framework, independent of any simulation tool, UrbiSim is made to incorporate real-world data and extended to cover many simulation use-cases including pedestrian and vehicular mobility. As the user behavior model might be based on membership of several social groups, it is also a powerful tool to investigate the impact of social interactions on ad hoc networks.**

I. Introduction

The final objective of protocol design, and more generally all ongoing research in computer networks, is to provide real-world users better performing and more reliable applications to use over a given network. However, it is still complicated to perform extensive experimentation on large networks under a controlled environment. Therefore, ad hoc network research heavily relies on network simulation in order to design, assess and improve communication protocols in this challenging context.

Network simulation requires a preliminary modeling effort, in order to capture the significant parameters of the environment that will directly impact the network characteristics and the performances of the applied protocols. Some of the main factors conditioning the network are the propagation model (which impacts the static topology of the network), the node mobility model (which impacts the dynamic topology of the network) and the traffic model (which impacts the dynamic load of the network and its user-perceived performance).

This task is complex to achieve as the captured environment should provide a trade-off between accuracy and relevance of the results and complexity so as to limit the simulation time. Therefore, a proper characterization of what aspects to capture in a model is deemed helpful.

A straightforward insight in a *simulation model* is to consider the set of factors that are impacting the network characteristics and performances. Those factors can be separated into two parts: on one side the *environmental factors*, on the other side the *user-dependent factors*.

Environmental factors account for whatever is possible doing during the simulation. This includes the propagation model which, based on the distance between users and other refinements, defines whether or not direct communication is possible. Another important feature is the set of constraints applied to users mobility, like restraining movements to streets, limiting the maximal speed or the motion evolution to respect some physical constraints.

User-dependent factors describe whatever is actually done in the simulation span. That is for instance the destination and paths that the users are actually taking, respecting the environmental constraints. This is also the data exchange pattern which depends on the user requests. Those user-dependent factors, to be accurately described, should include the social aspects that may determine the behavior and interactions.

As the traffic generated by users is also application-dependent, many simulation models, including this contribution, leave this question open to researchers focusing on application-layer problems.

The rest of this paper is organized as follows. In the next section, we will present the rationale for this contribution, which relies on an analysis of the current achievements and limitations in simulation models, detailed in section III. Then, we introduce the main concepts of our simulation framework, before going into a thorough description of its design and implementation in section V. Finally, future improvements of the model will be presented, before concluding on the current state of the development.

II. Rationale for a new simulation model

Current research showed that the social behavior of users has an important impact on network characteristics [10] and made this aspect as new open field for ad hoc protocols improvements, while considering the nature human interactions in design.

However, the majority of the emerging work on simulation models taking into account user behavior is either based on a custom simulator or is a simple proof of concept [19] [20]. In our contribution, we propose a framework that can be seen as an extension of industry or academic standards such as OPNET [5] or NS-2 [2], making this simulation model a *plugin* for existing simulators.

Besides, the urban environment is an excellent context of application for ad hoc networks, either to provide spontaneous electronically-enhanced interactions between users or to create networks of intelligent vehicles to improve safety while driving, or control flows of cars within a city; two use-cases that can be simulated using UrbiSim.

III. Overview of current simulation models

To reproduce the behavior of real users, simulation models can either be based on real-world traces or use algorithms with a random component to ensure variety of results [9]. While *trace-based models* replay data about mobility and/or connectivity previously recorded in a real-world experiment, *synthetic models* try to reproduce realistic behavior with a stochastic process.

Due to the lack of extensive data from the real-world, most research works rely on synthetic models, despite the risk of inaccurate conclusions on the assessed performances [22]. Besides, the most simple synthetic models make really simple and unrealistic assertions about the radio-waves propagation.

Some recent efforts tried to tackle this issue by proposing more advanced models by integrating the presence of buildings in mobility and propagation modeling [16] [14], by using available data from real world experiments to generate a mobility model [18] [17] or by using city maps to condition user movements [21]. Finally, some *dedicated simulators* attempt to reproduce an given use case with a more realistic approach.

A. Synthetic models

1) Simple synthetic models: Those models are the simplest attempts to capture user behavior in wireless networks. They assume that neither mobility nor propagation are constrained by any obstacle and sometimes break some simple rules of Newtonian's physics. Despite those limitations, they remain widely used in simulation, as they provide a fast way to setup an experimentation, and may serve as basis for more elaborated models.

The simplest of those mobility algorithms is *Random Walk*, a memoryless pattern similar to Brownian motion, where each user independently selects at random an angle and a speed for motion and changes after a constant time or distance interval.

Many variations on this model exist, the most widely used being *Random Waypoint* where, instead of an angle, an intermediate destination is chosen, and speed is selected within a given range. This mobility algorithm also introduces a random pause time at each intermediate destination.

2) Advanced synthetic models: To cope with the unrealistic speed and direction changes of the previous models, more advanced synthetic models have emerged.

The *Gauss-Markov Mobility Model* uses the memory of the $(n-1)^{st}$ velocities and directions to avoid sudden stops and sharp turns and a tuning parameter α allows to manage the degree of randomness in movements.

The *City section Mobility Model* constrains movements along pre-defined streets to a destination on those streets while using the shortest path and respecting speed limits and a distance with other nodes.

Bettsteter et al. also introduced the *Smooth Mobility Model* [7] [8] in which nodes have a set of preferred velocities and where an acceleration/deceleration threshold limits the speed change per unit of time. But, more interestingly, direction change (also managed by a stochastic process with a maximum angle change per unit of time) is there correlated with speed change, so that users slow down while turning like most drivers would do.

3) Environment-aware simulation models: In [16], the authors present the *Obstacle Mobility Model* which allows to insert buildings inside a simulation area. Those buildings will act as potential destinations for the mobile nodes, determine their pathways and have an impact on wireless transmission. Using the description of the building size and place, defined as an ordered list of corners coordinates, the model defines pathways using the Voronoi Graph technique [11].

However, the mobility model of this contribution seems quite limited as it in fact only offer a modified version of Random Waypoint along the paths defined by the Veronoi graph. No special attention has been ported neither to the fact that the destination building choice is not purely random, nor to define some group mobility that could mimic behavior of students and faculty inside a campus.

4) Social-aware simulation models: In those models, intermediate destinations chosen by the users are not purely random, but reflect an inner social behavior.

For instance, the Human Mobility model, introduced by Hogie et al. [15], tries to capture the behavior of users moving from a point of interest to another and stopping for a certain time at each point before moving to the next one. The points of interests also have an impact on the radio-waves propagation as they act as rooms with walls of different thickness.

B. Trace-based simulation models

In [18], Koberstein et al. introduce a *Graph-based mobility and propagation model* for simulation in an urban environment. Based on GPS data or maps from the OpenStreetMap [4] project, they extract a graph $G(V, E)$ representing the city section in which the nodes will evolve: the vertices V being the intersections and road bends, the edges E being the road sections. This model is built on the OMNeT++ [3] network simulator.

Parameters of each graph component allow to store waiting probability, average waiting time and the associated standard deviation on vertices, average speed and associated standard deviation on each edge. In this model, nodes motion is determined by a succession or random choices: choice of the next edge, of the speed, of the waiting time at a vertex.

The corresponding propagation model uses the direction created by the edges to generate two cones of preferred propagation (parallel to the edge) and two cones of reduced propagation (orthogonal to the edges) to account for the building obstructing nodes in different streets.

While taking into account the environment for propagation and mobility modeling, this model still relies on a probabilistic approach to govern the nodes movements. Besides, nothing is

said about the inner behavior of the nodes when the have to change speed and/or direction. Finally, this model does not capture any social aspect (like points of interest inside the city to which nodes are more likely to converge). This aspect could be used to generate a group component to the mobility model.

C. Dedicated simulators

Whereas most commercial or public source network simulators remain generic and leave the researcher implement more realistic models, others are dedicated to a given application case [13].

For instance, VanetMobiSim [12] has been developed for research on vehicular ad hoc networks and offer a fine simulation of the car movements, including management of overtaking and traffic signs, in an environment based on road topology read from pre-existing maps or randomly generated.

Although appearing redundant, this contribution and the one presented here can be considered complementary since VanetMobiSim does not offer a large-scale component to user mobility that would be governed by social aspects.

IV. Principles of the Urban Simulation Model

So as to overcome the shortcomings of the previously described models, UrbiSim is designed to simulate the behaviour of human *users* with communicating devices in a urban environment, represented by a set of *pathways* (straight portions streets, roads, etc.) and *spots* (buildings, parks, meeting points, etc.). For sake of realism in both the physical and sociological aspects, the mobility of users is modeled at two different levels: at small-scale for local speed and direction control, at a larger scale in order to assign meaningful intermediate destinations based on a user profile. We will now present in further details the components of our model: the urban environment, the user profile and the mobility management at large and small scales.

A. Urban environment

The urban environment used for this model consists in a set of spots linked together by different interconnected pathways. It can be generated stochastically using the Voronoi graph technique, as a section is shown on Fig. 1 or might rely on an external source (like OpenStreetMap [4] which uses a similar way to describe its maps using XML).

The set of pathways is stored as a weighted directed graph, where the weight of an arc is the expected time that may be required to traverse this arc. When a node wants to reach a distant spot, it simply runs a shortest-path algorithm on the stored graph from the spot at which it is currently located. For this computation, two modes are proposed:

In the first mode, the weight of each graph is simply the product of its length by the maximum speed allowed. This behavior is similar to a Global Positioning System (GPS) device planning for the shortest trip in time.

In the second mode, the traversal time is not fixed during the whole simulation. It is now also dependent on dynamic

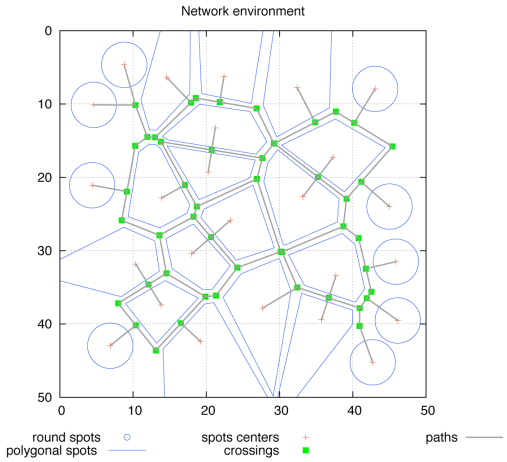

Fig. 1. Urban environment stochastically generated with UrbiSim

parameters like the current congestion of the pathway on this direction: the more cars are currently on this arc, the lower the maximum reachable speed will be. As those value may evolve, the path chosen by a user is recomputed at each intersection. This behavior is similar to what can be observed with a GPS device aware of traffic information.

B. User profile

The major contribution of this framework is also to introduce a way to model the behavior of human users in their everyday life using the concept of *social groups*.

As a configuration of the model, a set of social groups might be defined, which will set at which spot and at which time members of the groups should go and how long they should stay inside this spot. Then, each simulated could be assigned a *profile*, composed of the list of groups it belongs to. This allows to easily model the fact that some users belong to the employees of a given company, meet a group of friends at given time and place, or registered to the same gym class after work.

When users are found in a period of time when no destination spot is imposed by their profile, or if no profile is defined, a more naive random weighted choice of preferred spots is used to simulate a human spontaneously going to some of its favorite places.

This concept of user profile allows to study the impact of an heterogenous mobility pattern, alternating between transfers from a spot to another and more static stays within a spot, with different time-scales of interaction with users met more or less frequently.

C. Large-scale mobility

Large-scale mobility governs the way a user will globally move during the simulation time, using a finite-state machine (FSM) paradigm. In the current implementation, large-scale mobility is ruled by the FSM presented in Fig. 2 and described below. At simulation startup, after initialization each user starts with a random pause time inside its original spot. After this waiting period (or if its profile requests to go to another spot),

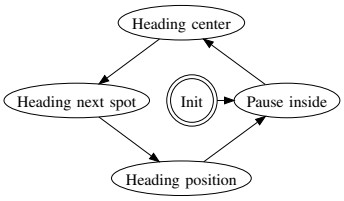

Fig. 2. Simple Large-scale mobility finite state machine

Fig. 3. Simple Small-scale mobility finite state machine

the user aims for the spot *center*, which is actually the doorway to exit the current spot. Once reached the exit of the current spot and joined to closest pathway, the user heads for its destination spot. During this phase, its mobility is managed by the *short-scale mobility* finite-state machine presented in section IV-D. When the destination spot is attained, the user enters and goes to a random place inside the building and starts another waiting period (set either by its profile or by the weight of this spot regarding this user). All movements within a spot are made under walking speed and currently follow the Random Walk mobility model.

D. Small-scale mobility

The model we present uses a separate finite-state machine that is responsible for preventing the user movements from being unrealistic at a small-scale, by preventing sharp turns and instantaneous speed changes. But in the current implementation, this automaton is also used to mimmic more closely the behavior of motorized vehicles in city streets.

As presented in IV-A, the urban environment is composed of a set of straight directed street sections, called pathways, each having a maximum reachable speed, either set statically (depending on traffic laws) or dynamically depending on the current congestion status of this section.

While entering a pathway, a user will progressively accelerate, trying to reach the current maximum achievable speed on the current street section, but it also computes the speed at which it should reach the intersection at the end of this pathway. This end-of-pathway or *target speed* depends on the nature of the intersection. If it is a simple road turn (only two pathways intersect) then the target speed is correlated with the angle the user will have to turn (bigger turning angle resulting in smaller target speed). On the contrary, if the intersection is a road crossing (more than two pathways intersect), then the user shall mark a stop and the target speed is set to zero.

Based on its breaking capacities, the current and the target speed, and its distance to the end-of-pathway intersection, the user might start to break. Depending on the pathway length, its current maximum speed, and the user characteristics, the breaking phase can occur before the maximum pathway speed is reached. An example of the evolution of a the instantaneous speed of a user is given in Fig. 4.

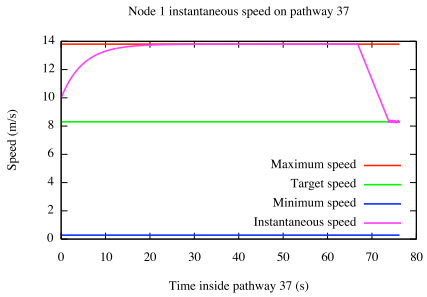

Fig. 4. Example of user speed variation inside a pathway with the polynomial-linear vehicle model

V. Framework implementation

Our simulation model is implemented as a C++ framework, trying to abstract from any simulator library. On the contrary, we based UrbiSim on standard an well renowned utilities, such as the Boost Graph Library [1] for all operations related to the graph abstraction of the urban environment.

Besides, configuration parameters and description of model elements, pathways, spots and users, rely on a XML representation for which a DTD specification is available.

By design, this contribution offers an API to interconnect with external network simulators. For the presented experiments, our framework has been tested with OMNeT++ [3] but it could, with little effort, be used with other supporting tools.

In this section, we will present the designed architecture of UrbiSim, depicted in Fig. 5, and how to interface it with host simulators and how to extend it.

A. Environment generation

The simulation environment is contained in a set of XML-based configuration files that store the definition and parameters of pathways, spots, groups and users.

The urban environment by itself, that is pathways and spots, is thought to be either stochastically generated or imported from the OpenStreetMap database. A PERL [6] script, external to the framework, allows an easy generation of pathways and spots, once given the size of the area to cover, the number of desired spots and other complementary parameters.

Based on this city definition, it is possible to create an additional configuration file setting social groups (with their associated spot and meeting period) and to assign each node a set of these groups in order to generate a complete profile.

Ideas of improvement include the creation of a similar script to stochastically generate groups and assignments while considering parameters from the real world, like assigning spots within the industrial zone to groups corresponding to co-workers going to the office, etc.

B. API for network simulators

Many simulators supporting ad hoc networks with mobile nodes share the particularity to mix a decentralized management of mobility and a centralized computation of the connectivity based on a propagation model: each node model

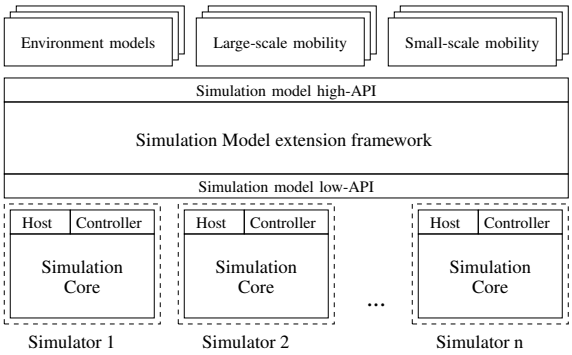

Fig. 5. Simulation model framework architecture

has a mobility component, and a controller object, with a global view of the network, dynamically creates or tears down communication links between the nodes.

We based the API of UrbiSim towards network simulators based on this observation and provide utility functions for both components: node mobility and network controller.

1) API for mobility component: To prevent computational overhead, network simulators update node positions on a regular interval. Hence, the basic function of a mobility component is to provide the position of the node at each interval, based on the adopted mobility model.

Using, our framework, this information is provided by a single function call to our main class, making the integration with an existing network simulator only requiring the implementation of a dummy mobility component which will trigger this function call at every position update interval. In any case, this method shall remain transparent for all other layers (application, routing, channel access) that are implemented in the host simulator.

2) API for controller component: Based on the current position of nodes and a propagation model, the controller component is responsible for computing whether or not a communication can occur.

The management of a propagation model aware of the urban environment has been left for future work, but the interface between UrbiSim and controller components of host simulators has already been designed. In future releases, this would allow to bypass connectivity computation inside the host simulator and to return a result based on the propagation model used in our framework.

C. Framework extension

Besides providing simple interface with host simulators, UrbiSim has been designed to allow easy extension of its features through the development of a consistent set of environment and mobility classes to manage additional parameters and events within the model.

1) Environment model: The environment model is responsible for constraining the mobility of users, that is using pathways in our current implementation, to provide a set of possible intermediate destinations, the spots or the intersections. It also describes the behavior model of users, that will govern the set of spots they are likely to visit.

More complete environments can be elaborated by enriching the definition of each of the component of the environment model. This can be done by adding parameters in the framework configuration files describing those components and extending the base class for those component to implement algorithms using the newly-defined parameters.

2) Mobility models: As presented in section IV, both small and large scales of user mobility are managed by finite-state machines respectively providing an sequence of intermediate destinations (may it be the position of an end-of-pathway intersection or a position inside a spot) and the position of the user at the next update interval. Functions providing this information are made virtual in mobility base classes

For instance, in the current implementation, the sequence of intermediate destinations is given by the shortest path algorithm, finding the fastest set of pathways to the chosen next spot; the node position at next update interval is computed thanks to the small-scale mobility finite-state machine.

Using the object-oriented nature of C++, it is fairly easy to implement more complex automatons or use any other paradigm to simulate behavior of users, as long as the same information about intermediate destination and next position is provided. This is simply done by extending the base mobility classes or the classes corresponding to the current implementation.

VI. Simulation results

UrbiSim has been tested with OMNeT++ [3] with multiple scenarios involving different urban environment (stochastically generated or extracted from OpenStreetMap [4]), and different number of users. For sake of example, figure 6 presents the mobility trace of a user during a simulated time of two hours using the urban environment depicted in fig. 1.

Simulation performance (ratio between simulated time and duration of the simulation execution) is dependent on the number of simulated users and refresh rate of the users postions. With our setup, we managed to simulate the evolution of hundreds of users with a reasonable simulation time using a standard desktop computer.

Finally, as this contribution only focuses on mobility, and because the associated propagation model remains to be developed (VII-B), the impact of our model on ad hoc protocols and network performances hasn't been investigated so far.

VII. Future work

In this section we will briefly present currently planned or investigated extensions to our simulation model.

A. Multimodal transportation

Currently, users are supposed to walk inside spots and to move using personal motorized vehicles to go from a spot to another. Adding the possibility for users to walk also to nearby spots could enhance the realism of our model.

Besides this simple improvement, we think possible to extend our contribution with the management of public transportation. Using the extensible description format used to define the urban environment, some pathways can be tagged

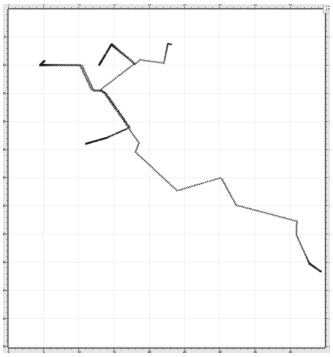

Fig. 6. Node mobility trace in environment presented in fig. 1

to be used only by special class of vehicles, hence creating railways and dedicated bus lines. In addition, particular spots would be created to account for train stations and bus stops. They will be visited by the public transportation vehicles and also by simple users waiting for the spot to be served.

This extension will also require a revision of the shortest-path algorithm to allow authorized users to integrate those new transportation means in their journey, while taking into account their speed and frequency of service.

B. Propagation model

However some commercial network simulators can provide an accurate radio-wave propagation computation, most tools used in academia only rely on simple models like free space. As an independent framework, UrbiSim should not depend on the simulator for this computation but be able to use it whether the provided model is considers accurate enough.

Therefore, our will is to provide a propagation model able to reproduce the impact of the many buildings present in an urban environment without being too computationally costly. The trade-off we propose is to attribute each environment area included between pathways an attenuation factor that will increase the fading effect as long as the signal traverses this area. Different attenuation factors could account for high buildings, suburbia or parks within the city.

As we focused on the mobility aspects, this important but distinct aspect of our model has been left for further implementation.

VIII. Conclusion

After considering the importance of an accurate simulation of the environmental and human-based factors impacting delay tolerant ad hoc networks and reviewing the shortcomings of the current contributions in that field, we presented UrbiSim, a framework for simulation within a urban environment with user behavioral model.

This model, by finely taking into account the sociological incentives of human urban movements, allows to more accurately investigate the short-term and longer-term interactions in an heterogenous mobility context, where real-world users alternate between extended stays at well-established locations and transition between those places. This can make this contribution a valuable tool for studies on mobile users communities.

Conceived as a framework, UrbiSim might be used as a plugin for many popular network simulators. Its extensible design allows to implement additional models for the environment or both scales of mobility, for instance to benefit from previously-existing models dedicated to a particular use case.

The formalism used to define the environment elements also make possible to add properties to those entities and to use them in redefined versions of the algorithms or automatons governing the behavior of users within the model.

References

[1] "Boost graph library," http://www.boost.org/.

[2] "The network simulator ns-2," http://www.isi.edu/nsnam/ns/.

[3] "Omnet++ discrete event simulation system," http://www.omnetpp.org/.

[4] "Openstreetmap project," http://www.openstreetmap.org/.

[5] "Opnet modeler," http://www.opnet.com/.

[6] "Practical extraction and reporting language," http://www.perl.org/.

[7] C. Bettstetter, "Mobility modeling in wireless networks: categorization, smooth movement, and border effects," *Mobile Computing and Communications Review*, vol. 5, no. 3, pp. 55–66, 2001.

[8] ——, "Smooth is better than sharp: a random mobility model for simulation of wireless networks," in *MSWiM*, 2001, pp. 19–27.

[9] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, pp. 483–502, 2002.

[10] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on opportunistic forwarding algorithms," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, p. 606, 2007.

[11] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*. Springer Verlag, 2000.

[12] J. Härri, F. Filali, C. Bonnet, and M. Fiore, "Vanetmobisim: generating realistic mobility patterns for vanets," *Proceedings of the 3rd international workshop on Vehicular ...*, Jan 2006.

[13] L. Hogie, P. Bouvry, and F. Guinand, "An overview of manets simulation," *Electr. Notes Theor. Comput. Sci.*, vol. 150, no. 1, pp. 81–101, 2006.

[14] L. Hogie, P. Bouvry, M. Seredynski, and F. Guinand, "A bandwidth-efficient broadcasting protocol for mobile multi-hop ad hoc networks," in *ICN/ICONS/MCL*, 2006, p. 71.

[15] L. Hogie, F. Guinand, and P. Bouvry, "A heuristic for efficient broadcasting in the metropolitan ad hoc networks," in *KES*, 2004, pp. 727–733.

[16] A. Jardosh and E. M. Belding-royer, "Towards realistic mobility models for mobile ad hoc networks," 2003, pp. 217–229.

[17] M. Kim and D. Kotz, "Extracting a mobility model from real user traces," in *Proceedings of the 25th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM06*, 2006.

[18] J. Koberstein, H. Peters, and N. Luttenberger, "Graph-based mobility model for urban areas fueled with real world datasets," in *Simutools '08: Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, pp. 1–8.

[19] M. Musolesi, S. Hailes, and C. Mascolo, "An ad hoc mobility model founded on social network theory," in *MSWiM*, 2004, pp. 20–24.

[20] M. Musolesi and C. Mascolo, "A Community based Mobility Model for Ad Hoc Network Research," in *Proceedings of the 2nd ACM/SIGMOBILE International Workshop on Multi-hop Ad Hoc Networks: from theory to reality (REALMAN'06)*. ACM Press, May 2006.

[21] P. Ruiz, B. Dorronsoro, D. Khadraoui, P. Bouvry, and L. Tardón, "Bodyf – a parameterless broadcasting protocol over dynamic forest," in *Workshop on Optimization Issues in Grid and Parallel Computing Environments, part of the High Performance Computing and Simulation Conference (HPCS)*, Nicosia, Cyprus, 2008, pp. 297–303.

[22] J. Yoon, M. Liu, and B. Noble, "Random waypoint considered harmful," in *INFOCOM*, 2003.