Graph-Based Mobility Model for Urban Areas Fueled with Real World Datasets

Jochen Koberstein, Hagen Peters, Norbert Luttenberger
Communication Systems Research Group
Dept. of Computer Science
University of Kiel, Germany
{jko|hap|nl}@informatik.uni-kiel.de

ABSTRACT

Mobile ad-hoc networks (MANETs) and especially mobile Wireless Sensor Networks (mWSNs) are embedded in the environment and therefore stand under strong influence of its specific characteristics. Beside e.g. sensor input, nodes motion patterns are supposed to be a very basic factor regarding performance. Hence simulations may need to account scenario specific mobility patterns while keeping the tradeoff related to simulation complexity in mind.

This contribution proposes a graph based mobility model, designed to resemble probabilistic node movements according to real world node paths like they may be induced by road grids. The model is presented along with a real world mWSN sample deployment from which the paths are extracted and against which the simulation fine-tuned.

Keywords

mobility model, wireless sensor networks, simulation, real world experiment, road grid

1. INTRODUCTION

In mobile networks like e.g. MANETs or mWSNs, network nodes move around in an operations area and yet stay in contact with a variable number of other network nodes—a number that depends on the relative position of nodes, their radio capabilities, and the distribution of obstacles, both to movement and to radio propagation. When a network simulator is used to investigate the behavior of such mobile networks, the network simulator should comprise according models to enable a realistic prediction on how the mobile network reacts to variations in the position of nodes and to resulting changes in the network topology.

In this paper we discuss a specialized type of graph-based mobility model and novel methods for instantiation. Mobility models are rule sets that allow a network simulator to generate trajectories for mobile nodes. "When a network simulator knows the position of a mobile node at any one time, the simulator can compute signal fading from one node to another and take actions based on the current network topology (e.g., determine the set of nodes that could receive a certain packet)" [17]. We introduce a simple accompanying radio propagation model that is especially suited for urban environments.

The mobility model has a major influence on the outcomes of simulation runs and their proximity to the "real" behavior of a mobile network [19, 1, 9]. Therefore, results obtained with unrealistic mobility models may not reflect the true performance of mobile networks in their operation areas. Many mobility models used today are artificial models, i.e. models that rely only on arbitrary decisions for next waypoints or movement directions and speeds. Prominent examples for this kind of models are random waypoint and random direction. These models are not related to special real world scenarios and often implicitly assume that real nodes can freely move over their operations area.

On the other end of the spectrum trace-driven simulation can be found. Although trace-driven simulation is typically closely related to the real world, it does not even approximately allow to explore the whole set of motion instances of the real world. Therefore a reasonable compromise between accuracy and artificiality is needed.

The mobility model we propose in this paper is a graph-based mobility model. Vertices of the graph are used to model geographic locations, edges are used to model paths between these locations. One such model is instantiated per network node that participates in a simulation run. Vertices carry additional attributes that comprise turning and pausing probabilities, edges carry additional attributes that are used to calculate node speeds. We show how a mobility model instance of this kind can be derived automatically from an initial GPS trace that is recorded in the operations area, or from map data. We also show how a GPS trace can be used in a similar manner for a trace-driven simulation. The latter enables us to employ trace-driven simulation for evaluation and fine-tuning of model-based simulation.

In the upcoming section we introduce the background of the conducted experiments and simulations, followed by an overview (section 3) of the thereby used attenuation model and its parameterization process. In section 4 we will give a detailed description of the mobility model and the derivation process used to generate the graph instance from position traces. Section 5 presents the evaluations of our approach. We discuss the work related to this paper in section 6 and a conclusion and outlook is given in the following section.

OMNeT++ 2008 March 3, 2008, Marseille, France

2. PREREQUISITES

2.1 Simulator Basis

Our approach is built upon the OMNeT++ simulation framework [18] accompanied by the Mobility Framework [3] and the SEE framework [4] (***S**imulator **E**xtension for Operations **E**nvironment Models*).

While OMNeT++ provides basic distributed event simulation the Mobility Framework adds basic node mobility and radio simulation. The SEE approach adds an operations area model and a hardware abstraction layer (HAL). The former provides meaningful sensor input to the WSN under test which is essential especially for data centric applications. The latter is based on the so called *xml-based Hardware Description Language* (xHDL, [4]) which provides for the possibility to describe the envisioned platform in a formal way and generate source code for different purposes—in particular the API which represents the HAL. This HAL enables portage of WSN application sources between simulation and hardware platforms in a seemless manner.

2.2 Sample Application Scenario

To gain experience regarding mWSNs in real world scenarios and to build some kind of reference for our simulations, we set up the so called WiSeBEES experiment (*Wireless Sensor-node Based Environment Exploring Swarm*). In this experiment mobile nodes operate in an urban operations area, collect sensor readings and share them with other nodes. The nodes are mounted on cars driving randomly within a dedicated urban operations area which is the same for all nodes. Thus the motion of nodes are induced by the road map of the area which implies very special connectivity patterns and spurious effects of the surrounding buildings on radio propagation. In the conducted field tests approximately 2 nodes per square kilometer are deployed. This results in a very sparse network with a highly dynamic topology where network partitions are the default.

The WiSeBEES nodes cooperate according to the so called *distributed virtual Shared Information Space* (dvSIS) paradigm [8]. Nodes are spreading out information throughout the network in a pro-active manner—or from another point of view: each node builds its own local view of the operations area by sampling, sending out and receiving data. The idea is to decide locally which data is valuable enough to be broadcasted to other nodes which are in radio range. E.g. older data is less valuable than newly sampled data, or if a node A has recently received data from node B, node A might suppose that B is still in range and B is not interested in data sampled by itself, thus A classifies data sampled by B as less valuable at the moment. This kind of controlled flooding scales quite well especially in highly dynamic topologies.

The data exchanged by the WiSeBEES nodes consist of temperature readings accompanied by meta information like position, assumed position accuracy, current velocity, time and sampling node id. The local view therefore allows to build a temperature map covering the operations area. Additional information about node speeds provide an indication about traffic conditions. A vehicular ad-hoc network scenario to disseminate locally relevant alarms or information is conceivable to be based upon the dvSIS paradigm.

The sample WSN nodes are based on modified Linksys WRT WLAN-Routers running *OpenWRT* [12]. To make this platform usable for WSN applications it was extended by a GPS-Module and an I^2C bus currently interfacing a temperature sensor. The WRT provides enough ressources to capture traces of information regarding the nodes position, dvSIS status and communication activities on unit.

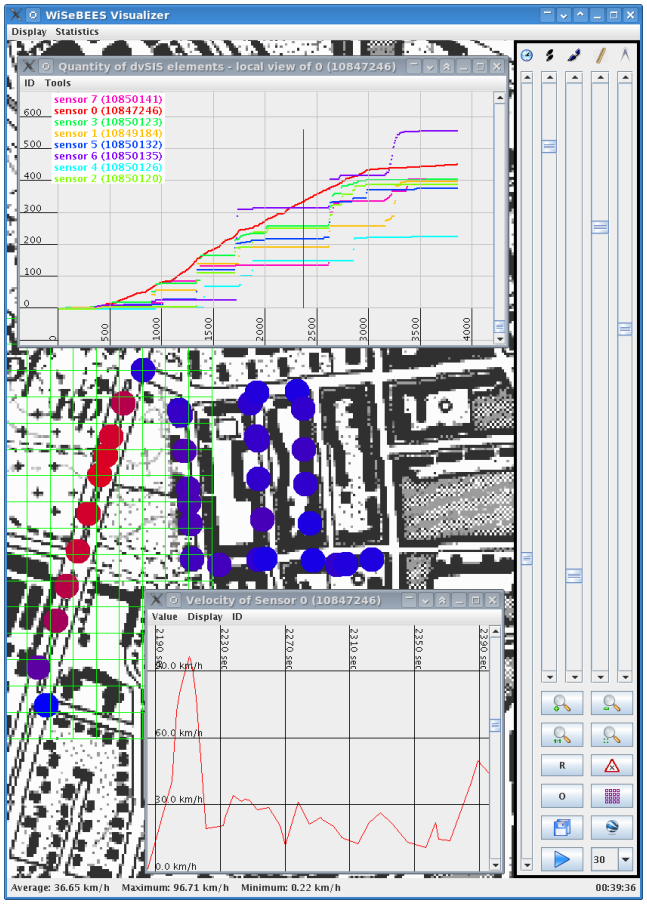

Figure 1: Offline visualization and analysis tool

2.3 Evaluation and Analysis Tool

To evaluate both, field tests and simulations, application instances perform quite extensive logging of actual sensor readings, position data, communication events with other nodes and node status. Each of these status traces (or logs) represents an execution—be it within a field test or a simulation—from a nodes' point of view. To evaluate several or even dozens of these logs and to build a global view from them, we have developed a visualization and analysis tool which provides for spatial and temporal playback of conducted executions. This tool also facilitates some statistical analysis and the export of nodes' position traces which are useful for further simulation purposes (see below). Figure 1 gives a glimpse on it: Sensor readings are projected on a map (in this example a node's velocity), sliders provide control over the temporal axis. Besides that, various display options and several statistical evaluations are supplied—in this example a node's velocity evolving over time and the growing number of readings collected in a node's local view.

3. MATCHING SIMULATION TO THE REAL WORLD

The log files collected during several WiSeBEES field tests (presented in the preceding paragraph) allow us to build more realistic simulation models for mWSNs and similar mobile networks. In the following section, we stepwise explain how especially radio attenuation model parameters can be tuned towards the real world scenario.

3.1 Attenuation Model

Common attenuation models for radio propagation range from the plain unit disc graph model to very sophisticated multi-path models. Since the latter ones need much processing power only for the attenuation model one needs to compromise between accuracy and usability. Our field test evaluations show that a plain unit disc graph model is not adequate for WSNs operating in an urban environment. Many factors influence the radiowave propagation, and it is impossible to take all of them into account. Nevertheless it turned out that there is a heavy influence by buildings—which is certainly not surprising. E.g. nodes may move in parallel along two parallel streets being "in range"—according to a unit disc graph model, but no communication takes place in the field test.

To keep the model as simple as possible we assume that the paths in the simulation are accompanied by streets of houses. Thus the radio attenuation tends to be lower along the path a node is currently traveling on and higher in the orthogonal direction. This is approximated using two cones of preferred propagation, i.e. reduced attenuation along the path, and two cones of reduced propagation, i.e. elevated attenuation. In most cases this model works pretty well as depicted in figure 2 and 3. Although in some rare cases like depicted in figure 4 it may lead to false positive assumptions.

This model of preferred and reduced propagation is combined with a randomized unit disc model leading to a randomized distorted unit disc model. Obviously more precise models based on the map are imaginable. The charm of this model is on the one hand the small extra costs regarding consumed processing power and on the other hand this propagation model could be easily implemented by using a slightly modified version of the `Nic80211` module that comes with the Mobility Framework.

3.2 Relating Parameters to the Real World

The original `Nic80211` module, simulating a uniform radio propagation, already provides many options for fine tuning, e.g. antenna sensitivity, transmitter power, thermal noise and so on. These are extended by the low and high attenuation parameters in the slightly modified version used here. To get as accurate as possible attenuation model parameters, we propose an iterative process to evolve the set of parameters as is depicted in figure 5.

In a first step, we extract waypoint traces (sequences of pairs of position and time) from the logs recorded in field tests. On the basis of the waypoint traces, and the analysis of communication events and the application specific data (see 2.3), we conduct the second step, the compare-simulate loop. In this loop we simulate the mWSN on basis of the waypoint traces and an initial set of attenuation model parameters. This trace-driven simulation is a close to "perfect" simulation of the mobile nodes positions (certainly depending on GPS accuracy and update frequency). The simulated mWSN also records log data (it runs the same application as the real world instance), which is analyzed and compared to the data recorded in the according field test. If the data does not match well (the relevant criteria are described below), the loop is executed again with an optimized set of attenuation model parameters. In case the log data recorded during the simulation run match the log data gathered during the field test well, the attenuation parameter set is assumed to be adequate.

One important criterion calculated from the recorded log data is the quantity of readings collected in a nodes local view (mentioned in 2.3), i.e. the number of sensor readings a node n of the mWSN "knows" at a time t. This quantity grows monotonically because n permanently samples new data[1], but if n meets another node m of the mWSN, the quantity increases abruptly, because n receives a large chunk of data from m and vice versa. Due to this unsteadiness, the curves reflecting the number of data elements nodes hold in their local views are a characteristic fingerprint of the current field test or simulation instance.

Figure 6 shows the match between the growth of the quantity of data elements gathered by a node in the field test and the same node in the simulation with an already fine-tuned attenuation parameter set. The remaining small deviation is caused by the "perfect" GPS and infinite hardware processing power found in the simulation.

A second criterion is the distribution of latencies that are incurred by data elements on their way through the network—or more precise: the distribution of durations between capturing a sensor reading and receiving it at another node. The shown histogram does not comprise the readings taken by a node itself since these are "received" immediately. The latency criterion is quite sensitive since it does not only matter how many readings are collected but also which readings and in which order. Already a single event of data exchange can lead to a significantly different picture. Figure 7 exemplarily depicts the distribution from the perspective of one node in the field test and the simulation. In reality about 250 data elements are received within the first minute after sampling them, in simulation the according readings count is about 4 % smaller. But again the behavior of the WSN in the field test and in the simulation match pretty well. The criteria chosen here are meant to be examples matching our scenario and may be exchanged.

[1]Assuming that nodes do not delete data elements from their local view.

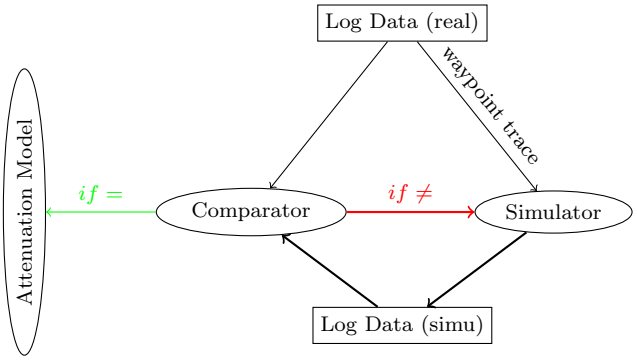

Figure 5: Tuning the attenuation model parameters.

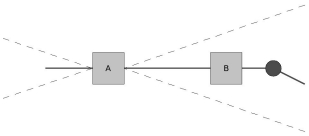

Figure 2: Attenuation along a path for nodes A and B

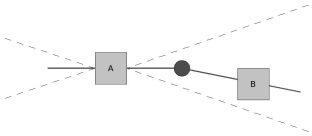

Figure 3: Attenuation across a bend for nodes A and B

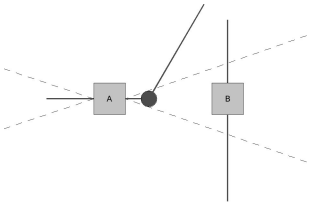

Figure 4: Wrong attenuation assumption

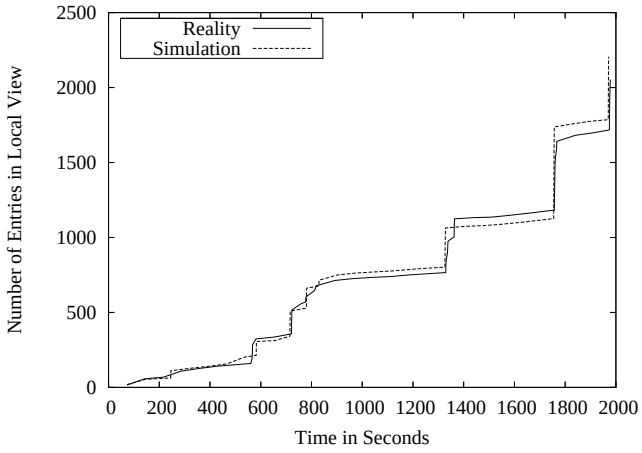

Figure 6: Sample growth of a nodes' local view in field test and simulation

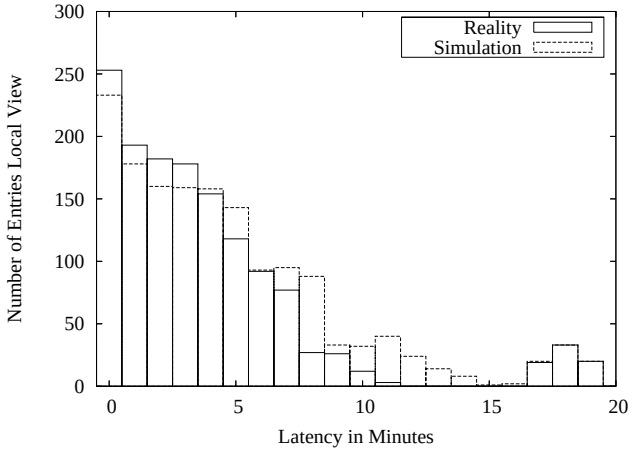

Figure 7: Field test vs. simulation regarding a latency distribution

4. MOBILITY MODEL

To resemble a road grid in a basic manner a mobility model instance needs to feature crossings, road bends, connections between crossings, waiting periods for nodes at crossings and meaningful node speeds.

The basis for the proposed mobility model is a directed graph $G(V, E)$. The vertices in the set V represent crossings and road bends while the edges in the set E represent roads connecting crossings and points of bends. Thus simulated nodes "move" along the edges. Edges have attributes that store the average speed and according standard deviation. Further attributes store the probability that an edge is selected by sensor nodes, located at incident vertices, to be the next edge to drive on. Vertices are configured by attributes that store their position and attributes that store the average waiting period and according standard deviation. An additional attribute holds the waiting probability on crossings.

The motion of a node is derived from a sequence of randomly made decisions concerning waiting periods, edges to drive on and the speed to drive. In the following, "edge" is sometimes used synonymous for elements in E and for straight lines between the positions of adjacent vertices.

4.1 Instance Generation based on GPS-data

The basis for graph instantiation is the data related to time and space recorded during field tests—namely measurement point respectively GPS-data sets. An algorithm to generate an entity of the mobility model needs to take the accuracy of this data into account, besides pure GPS inaccuracies e.g. interference of a nodes movement and position sampling rate may induce severe effects. Additionally such an algorithm has to be efficient in two different means: it has to generate an as accurate as possible model instance from an elementary field test and it needs to keep the model instance as compact as possible to reduce the consumption of processing power during simulation.

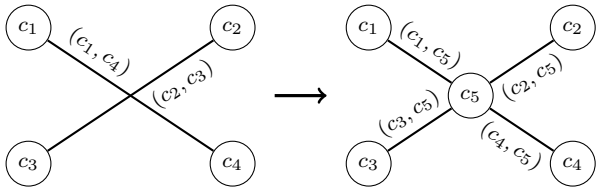

Figure 8: Detecting crossings (i.e. vertices) from intersecting edges.

This means to reduce the number of vertices and edges to a minimum without discarding relevant information. For example, two very closely located vertices may probably describe the same crossing. In this case the same information is stored twice. Also a vertex with only two adjacent edges and an angle of almost 180 degrees between these edges may be unnecessary, because a vertex with two edges represents a road bend where waiting probability is zero and probably not even u-turns are made.

The following paragraphs give an overview into the derivation process used in our approach. The first step generates a graph $G(V, E)$ where each measurement point (a pair of position and time) recorded during a field test is represented by a unique vertex in V and edges are set between ver-

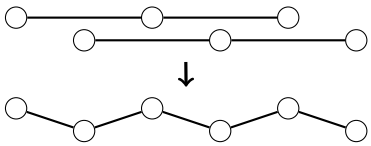

Figure 9: Merging edges describing the same street.

tices that represent chronological neighboring measurement points sampled by one node in the field test. In this state the created mobility model instance resembles multiple waypoint traces like described in 3.1.

The next step is to build a graph that has no intersecting edges. Therefore the algorithm first searches for intersections of edges and creates a new vertex with position at the interception point and updates E. This step is depicted in figure 8. The left graph shows the situation before, the right after this step. In this way additional crossing, i.e. vertices, are detected and the graph is freed from intersecting edges.

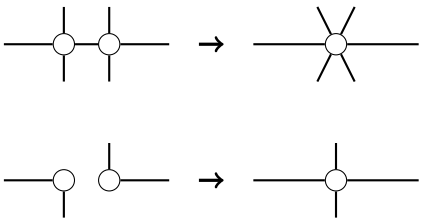

Figure 10: Merging vertices describing the same crossing.

The next step is to merge similar vertices and edges or even strip off unnecessary ones. Three important merge operations are used: First merging of paths which probably describe the same way as shown in Figure 9. Second, merging vertices with redundant information as depicted in the upper half of figure 10. And third, the detection of a crossing in reality by merging two vertices that are close together and describe two road bends as shown in the lower half of figure 10. Thereby the distance D between the positions before and after the merging operation of a vertex is bounded by a value significantly smaller than WLAN radio range.

Further more the recorded traces may by overly fine-grained and redundant vertices and edges are discarded as e.g. depicted in 11. A set of GPS-traces and the derived graph instance for the sample operations area referred to in 5 is shown in figure 12.

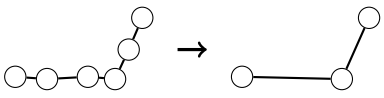

Figure 11: Deleting redundant vertices and edges.

The resulting graph matches reality well in the following sense:

The most precise (but without degrees of freedom) mapping of real motion to simulation available is the above mentioned trace-driven simulation. Assume a simulation, based on the same field test data, with two simulated nodes, A and B. Let A's mobility module be configured in "trace-driven" modus, let B's mobility module be a computed instance of the introduced mobility model.

Due to the fact that the graph of B's mobility model instance is derived from A's waypoint trace by merging and deleting some vertices and edges in a well-defined way, we can state two assertions:

1. If we locate A and B close to each other (i.e. with distance smaller than d) at the start of the simulation, we can find for any possible position of A a sequence of randomly made decisions (in the sense stated in 4) of B, so that the distance between A and B is smaller than distance d.

2. For any possible position B has, there is a position A has during simulation, such that the distance between A and B is smaller than distance d.

The value of d is bounded by D, so d is significantly smaller than the radio range and assuming that the inaccuracy of GPS measurements is small compared to the radio range too, a relatively close relation between the mobility model instance and the waypoint trace, and thereby real world scenarios, is established.

Figure 12: Set of measurement points (upper) and the thereof extracted graph (lower)

4.2 Instance Generation based on Map-data

Besides the generation of graph instances from real world GPS-data—which are so to say customized for the envisioned operations area—we allow for the usage of existing map data as basis. We chose the OpenStreetMap (OSM, [11]) project as source for these data sets for three reasons: First these data sets are freely available and are leveraged by an open (map-) source community. Second, it is accessible via simple HTTP requests. Thus data can either be retrieved online or be prepared and stored in advance. Third it is not limited to certain regions but available for the whole world.

The map data is created manually by members of the OSM community—similar to a wiki—and is available in an XML-based format, which is described by a DTD. The essential elements are *nodes* (not to be mixed up with nodes in the wireless network), describing points with tuples of latitude and longitude, *ways* connecting two or more nodes, and *attributes* which further characterize nodes and ways.

The generation of the graph of a mobility model instance is parted in four steps: First the map of a selected rectangular area is retrieved from the OSM servers and if applicable stored into a local file for future use. In the second step additional information beyond the road grid contained in the map, e.g. points of interest, parks or public facilities, need to be stripped off. The third step is the mapping of the stripped map data to a graph instance like described above. Nodes are mapped to vertices and ways are parted at the nodes and the sections are mapped to edges. The accompanying attributes define—among others—the type of road, e.g. primary road, residential road, cycleway or footway. These are mapped to according (but naturally user definable) average speed parameters of the graphs edges. Furthermore attributes may restrict ways, e.g. one-way streets result in an according unidirectional edge in the resulting graph. In a final step the generated graph is validated and optimized. This is necessary due to several reasons—to mention the most important: Dead ends throughout the graph, which leave no possibility of return (e.g. former one-way streets or simply faulty maps), are forced to be eliminated since nodes of the wireless network would possibly be trapped during execution. Furthermore cutting out a rectangular piece of the world typically leaves dead ends at the border of the selected area—these can be eliminated, too.

5. EVALUATION

To evaluate a generated and fine tuned mobility model instance, it would be preferable to compare real world results with results derived by simulation. To obtain statistically firm findings, a considerable amount of field tests and simulation runs need to be performed—while the latter are at most a matter of time and processing power, the former induce extremely high costs in different aspects. Thus we decided to leave it at the comparison done during the fine tuning cycles and instead compare the graph-based mobility model instance described above against a generic mobility model instance, namely random waypoint with a unit disc propagation model.

The graph-based mobility model instance based on GPS-data is derived from field tests conducted in the city of Kiel, Germany. As examples for the OSM-based instances we have selected equally sized cut-outs of two different cities,

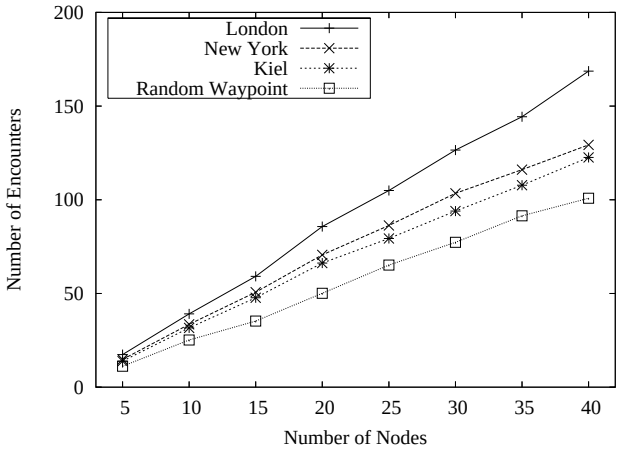

Figure 13: Number of node encounters

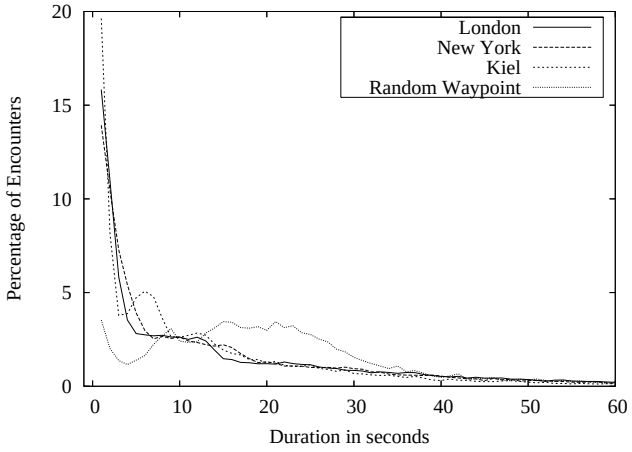

Figure 14: Distribution of encounter duration for simulation with 40 nodes

namely New York and London. While New York is characterized by the typical Manhatten grid, London is parted by the Thames and comprises a lot of one-way streets.

For all performed simulation runs parameters like playground size, operations area model, transmission power, simulation time, average node speed, implementation etc. are fixed. All evaluations assume a duration of one hour of simulated time. The number of nodes participating in the network is varied from 5 up to 40. Despite the comparisons described in 3.1 these evaluations are of a statistical nature and therefore depend on metrics beyond the "looks good" category.

No completely generic performance metric for mWSNs or MANETs exist. Hence common performance metrics are adapted to the scenario or even very application specific metrics are introduced. According to our WiSeBEES experiment we chose three metrics at different levels of complexity:

1. At a very low level we simply count the number of encounters between nodes, this is obviously related to the topology.

2. The medium level is represented by an average latency

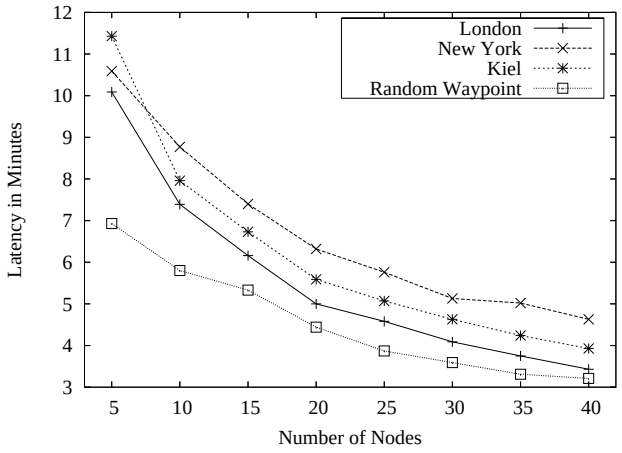

Figure 15: Average latency

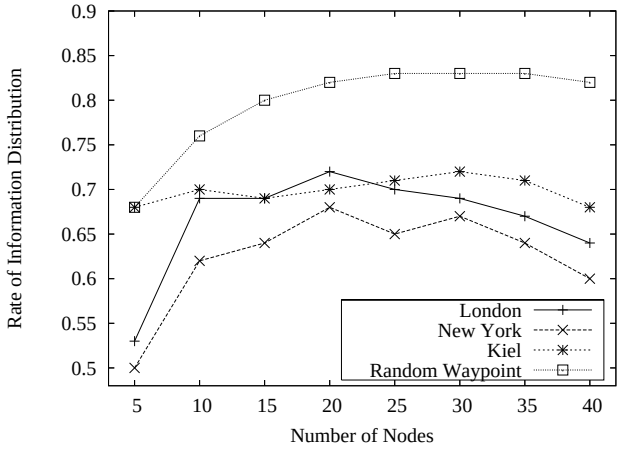

Figure 16: Average percentage of information a single node holds compared to the total sum of information throughout all nodes.

by means as described in 3.1. This is related to network performance.

3. The high level and very application specific metric is represented by the amount of information a single node has collected (number of readings in the local view of the dvSIS) compared to the whole set of information available throughout the network.

As stated above the partition of the WiSeBEES network is the default case. Therefore the number of encounters where nodes meet other nodes is an essential parameter for the performance of the WSN—more encounters imply more possibilities to exchange information. The inhomogeneous distribution and movement of nodes on dedicated paths encourage the possibility of encounters in contrast to the homogeneous distribution of nodes over the operations area using the random waypoint model assuming free space. This is reflected in the results depicted in figure 13 which shows the overall number of encounters within a simulation run over the number of participating nodes. Comparing the graph-based mobility model instance for London with the random waypoint model shows about 75% more encounters in the former case.

On the other hand the duration of encounters also varies significantly between the two models: Using the graph-based mobility model instances the number of very short encounters is high—nodes meet with opposite direction—and only some last very long—nodes move on the same path in the same direction. In contrast the random waypoint model instance allows for arbitrary encounter durations. Figure 14 shows that about one third of the encounters last less than 5 seconds using the graph-based model instances. Thus the graph-based mobility model instances lead to worse mWSN performance: Either the encounter lasts too short to exchange much information or the nodes eventually have already exchanged all information they have long before the encounter ends.

Accordingly the information dissemination is hampered using the graph based mobility instances, which is reflected in the latencies as depicted in figure 15. The voyage of readings last up to 80% longer using the graph-based mobility model instances compared to the random waypoint model instance. In all cases one can state a lower variance of the latency for scenarios with an increased number of nodes.

Figure 16 shows the percentage of the average amount of information a node holds in its local view compared to the total sum of information which is already captured by the mWSN as a whole. As in the case of the latencies the mWSNs performance is worse using the graph-based mobility model instances. Especially for more dense deployments this percentage differs significantly between the model instances.

These evaluations show significant differences on the one hand between the two underlying models but also between the graph-based model instances referring to different road grids. The concrete reason or roots for the latter ones are probably not to determine in detail, but this circumstance argues for the use of real world data in simulations to make predictions for an envisioned deployment more accurate.

6. RELATED WORK

Camp et al. [1] pointed out the importance of nodes' trajectories for mobile ad-hoc network behavior—they result in specific topology changes which can be verified even in an analytical form [9]. Many simple mobility models, still in use, have severe limitations as for the reproduction of real world scenarios [19]. Hence more sophisticated models have been proposed e.g. by Jardosh et al. [5], who add artificial obstacles in terms of mobility and radio propagation to the simulated operations area. Facilitating an inverse perspective Tian et al. [15] use a randomly generated graph to model paths that nodes are allowed to move along—similar to our approach. While this is supposed to produce more significant results it is not directly related to a real world deployment.

Since trace driven motion of nodes may be very closely related to the envisioned real world scenario it lacks degrees of freedom to enable the derivation of new trajectories needed to produce statistically firm results. An extraction process using real world position traces may lift the captured information to a higher level of abstraction: Kang et al. [6] and Patterson et al. [13] propose approaches for the extraction of places respectively ways from GPS traces recorded from a users life. Even so they do not aim for a generalized mobility model. In contrast Kim et al. [7] and Tuduce et al. [17] propose the extraction of paths viable for nodes from traces of

WLAN connectivity, which comes quite close to the issues of this contribution. Furthermore it does not depend on GPS hardware and signals which enables the usage e.g. in buildings. Nevertheless the existence of a WLAN infrastructure is inevitable and the effort stands in a disadvantageous relation to the gainable accuracy. The latter issue is taken up by Yoon et al. [20] who enrich the information of the coarse-grained wireless traces with an existing road map. Despite the gain in accuracy this selects so to say a subset of the map for the use in the mobility model of the simulator.

Using mobility models solely based on map data is typically found to be used in the field of vehicular ad-hoc network simulations. Some of these models go way deeper, down to the behavior of single traffic participants and their spacial interaction. Approaches based on the TIGER [16] data sets are e.g. discussed in [10, 2, 14]. Despite TIGER data sets OSM data sets are not limited to some regions of earth and are advanced using the open source philosophy.

7. CONCLUSION

The approach presented in this contribution seizes this suggestion to abandon purely artificial mobility models for wireless network simulation and proposes a fine-tuning and extraction process based on field tests.

Since the results are promising we plan to add a simplistic traffic jam extension to the presented mobility model. Furthermore the integration of the extraction process in a common editor for OSM data will enhance usability and allow a convenient adaptation of maps generated from GPS data.

8. REFERENCES

[1] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad-hoc network research. 2(5):483–502, 2002.

[2] D. R. Choffnes and F. E. Bustamante. An integrated mobility and traffic model for vehicular wireless networks. In *VANET '05: Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks*, pages 69–78. ACM, 2005.

[3] W. Drytkiewicz, S. Sroka, V. Handziski, A. Koepke, and H. Karl. A mobility framework for OMNeT++. In *3rd International OMNeT++ Workshop*, 2003.

[4] J. Koberstein and N. Luttenberger. System-Level WSN Application Software Test using Multi-Platform Hardware Abstraction Layers. In *The 2nd International Conference on Mobile Ad-hoc and Sensor Networks (MSN 2006)*, Hong Kong, 2006.

[5] A. Jardosh, E. M. Belding-Royer, K. C. Almeroth, and S. Suri. Towards realistic mobility models for mobile ad hoc networks. In *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 217–229. ACM, 2003.

[6] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello. Extracting places from traces of locations. In *WMASH '04: Proceedings of the 2nd ACM international workshop on Wireless mobile applications and services on WLAN hotspots*, pages 110–118, New York, NY, USA, 2004. ACM.

[7] M. Kim, D. Kotz, and S. Kim. Extracting a mobility model from real user traces. In *25th IEEE Conference on Computer Communications (INFOCOM)*, Barcelona, Spain, 2006. IEEE.

[8] J. Koberstein, F. Reuter, and N. Luttenberger. The XCast approach for content-based flooding control in distributed virtual shared information spaces - design and evaluation. In *1st European Workshop on Wireless Sensor Networks (EWSN)*, Berlin, 2004.

[9] T. K. Madsen, F. H. P. Fitzek, and R. Prasad. Impact of different mobility models on connectivity probability of a wireless ad hoc network. Oulu, Finland, 2004.

[10] R. Mangharam, D. S. Weller, D. D. Stancil, R. Rajkumar, and J. S. Parikh. GrooveSim: a topography-accurate simulator for geographic routing in vehicular networks. In *VANET '05: Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks*, pages 59–68. ACM, 2005.

[11] OpenStreetMap. *available at* `http://www.openstreetmap.org`.

[12] OpenWrt. *available at* `http://www.openwrt.org`.

[13] D. J. Patterson, L. Liao, K. Gajos, M. Collier, N. Livic, K. Olson, S. Wang, D. Fox, and H. A. Kautz. Opportunity knocks: A system to provide cognitive assistance with transportation services. In *6th International Conference of Ubiquitous Computing (UbiComp)*, volume 3205 of *Lecture Notes in Computer Science*, pages 433–450. Springer, 2004.

[14] A. K. Saha and D. B. Johnson. Modeling mobility for vehicular ad-hoc networks. In *VANET '04: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, 2004.

[15] J. Tian, J. Hähner, C. Becker, I. Stepanov, and K. Rothermel. Graph-based mobility model for mobile ad-hoc network simulation. In *35th Annual Simulation Symposium (ANSS-35)*, San Diego, California, USA, April 2002. IEEE Computer Society.

[16] Topologically integrated geographic encoding and referencing (TIGER), U.C. Bureau. *available at* `http://www.census.gov/geo/www/tiger/`.

[17] C. Tuduce and T. Gross. A mobility model based on WLAN traces and its validation. In *24th IEEE Conference on Computer Communications (INFOCOM)*, Miami, USA, 2005. IEEE.

[18] A. Varga. The OMNeT++ discrete event simulation system. In *Proceedings of the European Simulation Multiconference (ESM)*, 2001.

[19] J. Yoon, M. Liu, and B. Noble. Sound mobility models. In *The Ninth Annual International Conference on Mobile Computing and Networking*, pages 205–216. ACM, 2003.

[20] J. Yoon, B. D. Noble, M. Liu, and M. Kim. Building realistic mobility models from coarse-grained traces. In *MobiSys '06: Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 177–190, New York, NY, USA, 2006. ACM.