

# Emulation of Large Scale Wireless Sensor Networks: From Real Neighbors to Imaginary Destination

Bogdan Pavkovic, Jovan Radak, Nathalie Mitton, Franck Rousseau, Ivan Stojmenovic

▶ **To cite this version:**

Bogdan Pavkovic, Jovan Radak, Nathalie Mitton, Franck Rousseau, Ivan Stojmenovic. Emulation of Large Scale Wireless Sensor Networks: From Real Neighbors to Imaginary Destination. 11th International Conference on Ad-Hoc Networks and Wireless, Jul 2012, Belgrade, Serbia. hal-00686693

**HAL Id: hal-00686693**
**https://inria.hal.science/hal-00686693**

Submitted on 11 Jul 2012

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# From Real Neighbors to Imaginary Destination: Emulation of Large Scale Wireless Sensor Networks

Bogdan Pavkovic[1], Jovan Radak[2], Nathalie Mitton[2], Franck Rousseau[1], and Ivan Stojmenovic[3]

[1] Grenoble Informatics Laboratory (LIG), University of Grenoble, France
{firstname.lastname}@imag.fr,
[2] INRIA Lille - Nord Europe, France
{firstname.lastname}@inria.fr,
[3] SITE, University of Ottawa, Canada
ivan@site.uottawa.ca

**Abstract.** The ultimate test for many network layer protocols designed for wireless sensor networks would be to run on a large scale testbed. However, setting up a real-world large scale wireless sensor network (WSN) testbed requires access to a huge surface as well as extensive financial and human resources. Due to limited access to such infrastructures, the vast majority of existing theoretical and simulation studies in WSN are far from being validated in realistic environments. A more affordable approach is needed to provide preliminary insights on network protocol performances in large WSN. To replace large and expensive realistic testbeds, we introduce a novel approach to emulation. We propose a specifically designed experimental setup using a relatively small number of nodes forming a real one-hop neighborhood used to emulate any real WSN. The source node is a fixed sensor, and all other sensors are candidate forwarding neighbors towards a virtual destination. The source node achieves one forwarding step, then the virtual destination position and neighborhood are adjusted. The same source is used again to repeat the process. The main novelty is to spread available nodes regularly following a hexagonal pattern around the central node, used as the source, and selectively use subsets of the surrounding nodes at each step of the routing process to provide the desired density and achieve changes in configurations. Compared to real testbeds, our proposition has the advantages of emulating networks with any desired node distribution and densities, which may not be possible in a small scale implementation, and of unbounded scalability since we can emulate networks with an arbitrary number of nodes. Finally, our approach can emulate networks of various shapes, possibly with holes and obstacles. It can also emulate recovery mode in geographic routing, which appears impossible with any existing approach.

**Keywords:** emulation, simulation, routing, wireless sensor networks.

## 1 Introduction

A plethora of theoretical results and practical applications have emerged from the wireless sensor networks (WSN) research domain. One of the main issues in WSN is experimentation on real testbeds. The vast majority of existing testbeds consist of several dozens of sensor nodes. They satisfy the need for experimenting with centralized algorithms in small scale deployments. However experimental evaluation of network layer protocols intended for large scale WSN is still unfordable to most researchers due to several issues. The cost of buying a large number of sensors to start with, most researchers normally do not have such resources. Then the need to deploy them physically in vast environments. Finally, providing appropriate human resources for their maintenance. Thus, the most popular way for validating and comparing algorithms and solutions is through carefully driven simulation [15], using different types of software simulators such as ns-2, OPNET, WSNet. There is a handful number of existing large scale sensor network deployments, such as Senslab [13], Wisebed [20], GreenOrbs [8], allowing researchers to test their solutions in a real environment. Using these testbeds for validation and comparison of protocols raises two main challenges: *scalability* (how protocols perform on larger networks?), and *pattern* (sensors are usually placed in a regular structure and with certain density which limits possible patterns for investigation).

A compromise between simulation and real testbeds is emulation. Emulation combines elements of real environment experimentations with some assumptions normally taken during simulations. It generally has realistic parameters which are directly incorporated (by software) into the architecture being used [14, 17, 6]. Sometimes all nodes are real. Virtual nodes might also be added [3]. In Wisebed project [20], simulated node corresponds to either existing or virtual node. Links between virtual nodes, between a real and a virtual node, and even links between two real nodes are simulated (parameters are sent to a base station that makes simulated decisions and returns communication results back to the real participating nodes) [20].

A solution based on using smaller networks to emulate the behavior of the large scale networks has been proposed [9]. Up to 50 sensors, all within a 1-hop neighborhood, are deployed. The real source node is placed at the center of a real 1-hop neighborhood, while the destination is virtual and placed outside this neighborhood. The process of emulation is performed as follows. *(i)* The message is being held at the source node $S$, all available nodes serve as actual one-hop neighbors, and $S$ chooses the best forwarding node $B$ among them, according to the routing algorithm. *(ii)* Node $B$ is remapped to source node $S$ and the position of the virtual destination is recalculated, translating it by the vector $-\overrightarrow{BS}$. In order to provide a realistic variability of the signal propagation conditions, the virtual destination is also rotated around $S$. The goal is to change the set of potential forwarders experienced by the source node $S$ between the successive steps as in real routing. We will refer later in the text to this as randomness of 1-hop neighborhood. *(iii)* Node $S$ is again source node (after remapping from

node $B$) and the process is repeated until the virtual destination falls within 1-hop neighborhood or the routing algorithm fails.

With this setup, we can provide realistic and accurate results using real sensor nodes and real wireless links among them, while providing scalability for experimentation scenarios with a virtually unlimited size of the emulated network. Each sensor is provided with its own geographic location (by software following actual measurements), and accurate location for all of its neighboring sensors. The local neighborhood of the source node is fixed and a subset of the same nodes are forwarding candidates at each hop. Emulation is only due to enforced mapping of large set of virtual nodes to a small set of real nodes.

In this paper we propose a different approach to the emulation of large scale wireless sensor networks. The main goal is to achieve better randomness of neighborhood structure, and control network density at the desired level. We use pseudo-randomness instead of full randomness in the location of neighbors. Our 43 nodes are deployed in a hexagonal pattern. We also "return" the message from the forwarding neighbor back to the initial source node by our additional software links, so that the next step may be carried out. However we do not need to rotate the virtual destination node. Instead, the original full size simulated network is translated so that the source node is the origin, and its neighbors from simulated network are all rounded to their nearest physical node from the actually deployed hexagonal network. Consequently, only a subset of the nodes forming the real hexagonal network are considered as forwarding candidates, and this set changes at every routing step. We have evaluated our system on greedy geographical algorithm (GARE) and cost over progress greedy algorithm (COP_GARE) [9] as well as XTC algorithm [19].

In Section 2 we give more details on emulation of sensor networks and related work. The description of our emulation setup with the details on the theoretical background are given in Section 3. We present results obtained with this emulation framework and evaluate their accuracy in Section 4. Finally we summarize and give conclusions in Section 5.

## 2 Related work

The ultimate test for many protocols would be to run them on an existing real-world large scale wireless sensor network testbed [20, 13, 1]. The largest WSN testbed in the world is GreenOrbs which consists of 1 000+ deployed sensors [8]. Physical testbeds for WSN systems tend to be small in scale, expensive to maintain and time consuming to set up, mainly due to the huge amount of human resources needed [3]. They also lack in flexibility, offering only a fixed topology and limited heterogeneity and programmability. Repeatability is most of the time impossible since many relevant operating parameters are beyond user control.

### 2.1 Different approaches to emulation

Different approaches have been proposed for emulation so far. The main goal is to overcome the shortcomings of simulation while staying away from the complexity

of real world experimentation and providing a certain level of repeatability. It duplicates the functionality of one system in terms of another system, offers greater fidelity than simulation and greater flexibility than physical testbed. Nevertheless, in the literature, the term emulation has been endorsed by different approaches.

Emulation of large scale networks has been studied in a variety of contexts [2, 10, 5]. Few different types of approaches are proposed. The main goal in all approaches is to overcome the deficiencies of simple simulation either with environment emulation — in which the characteristics of the real nodes are built-in and executed in simulator — or using network emulation — in which each node communicates with a real node in order to obtain more accurate results. Fall [4] distinguishes network emulation — where real traffic is exchanged between real and simulated nodes — and environment emulation — where real implementations are embedded in a controlled environment.

One of the first definitions of emulation was proposed by Ke *et al.* [7] by using a combination of simulated and real sensors. They have added three simulator modules, real time scheduler, network objects and tap agents, to achieve better cohesion with the hardware and to better simulate it in the ns-2 simulator. Two real machines are attached to the endpoints of the network producing the traffic that is later injected to emulated nodes by the ns-2 server. Satisfactory results were obtained for the emulation of 10 to 120 mobile nodes. However the experiments could not be carried out in real time.

A similar concept can be found in SensorSim [12], where the authors propose hybrid simulations. They combine real sensor readings (geophone, microphone and infrared detector) with simulated nodes in order to get more realistic event detection in military scenarios.

Object-oriented representation of sensors, communication channels and physical media (mobility model, power model, etc.) has been used in J-Sim [14]. In this approach, a virtual simulation environment is integrated with a small number of real hardware devices to facilitate performance evaluation of real-life devices at a large scale. Application specific models are developed using an object-oriented model, subclassing the simulation framework. The environment is thus well controlled and hardly tunable to fit any other deployment context.

Physical layer emulation has been proposed using an FPGA based DSP engine [6]. RF signal propagation in a physical space can be emulated for a wide variety of wireless devices. This work focuses on channel emulation and provides satisfactory results for higher layer performance evaluation of real systems in a controlled propagation environment.

Coulson *et al.* [3] have described a virtual testbed model for flexible experimentation in WSN by seemingly integrating physical, simulated and emulated sensor nodes and radios in real time. It was also demonstrated as part of the WiseBed [20] project. Their main approach to emulation is to add simulated nodes, emulated nodes, virtual links between two nodes (each endpoint could be a real, emulated or simulated node) and support different inter-node connectivity patterns in a physical testbed. Virtual link between two physical nodes can

be created by connecting each of them to a base station, connected via Internet, and simulating the link performance in real time under different parameters from those currently present in the testbed. A base station connected to a physical node can similarly run a virtual link between a physical and a simulated node. However, the concept of emulated node and virtual links involving such nodes, has not been properly defined or exemplified.

### 2.2 Emulation using a real one-hop neighborhood

Our current emulation framework is based on an idea introduced in previous work [9]. A set of real nodes is placed randomly using the MIN-DPA algorithm [11] in a 1-hop neighborhood of the designated central node $S$, within a circle of radius $r$. The routing destination $D$ is situated further away from the 1-hop neighborhood, and is a virtual node. This setup is depicted in Fig. 1.

At each step, node $S$ chooses a new forwarding node (*e.g.* node $B$) according to the routing algorithm being used. Then the packet being routed is delivered to the selected next hop over the real radio link — with medium access, propagation and interference. When using greedy routing, the packet makes progress towards the virtual destination $D$, if at least one neighbor is closer to $D$ than $S$. The coordinates of the virtual destination $D$ are updated by $-\overrightarrow{SB}$ and correspondingly, node $B$, the new holder of the packet in transit, is translated to the original source node $S$. This translation allows to reuse the same physical nodes as forwarding candidates in the next step. In other words, the virtual destination $D$ can be considered closer, at the new position $D'$, to same 1-hop neighborhood, and mobile. The same procedure is repeated until the position of the virtual destination $D$ falls inside the 1-hop neighborhood, *i.e.* it can be reached directly. In this final step, routing towards the nearest physical neighbor to $D$ is performed.

One problem with this approach is that the layout and size of the 1-hop neighborhood of $S$ does not change between successive routing steps, which is not realistic. To overcome this to a certain extent, $D'$ is further rotated by a random angle to a new position $D''$ as shown in Fig. 1. Hence the actual candidate neighborhood subset is rotated at each step, while preserving the distance to the source. This results in more variability in the 1-hop neighborhood, providing more randomness in the emulated neighborhood along the path. Obviously, we cannot expect full randomness of the 1-hop neighborhood with such a rotation.

The other problem with this emulation is that the neighborhood density is fixed at the very beginning, at deployment stage, and is not changed or controlled later during emulation. Further, while rotating the virtual destination increases the randomness of the neighborhood, it does not allow to control an experiment with obstacles and deal with sparse networks. When no neighbor is closer to destination than the current node, recovery mode is called upon. Recovery mode cannot be emulated with this approach, because the rotation of the neighborhood will interfere with the mandatory traversal of particular faces of the Gabriel graph.

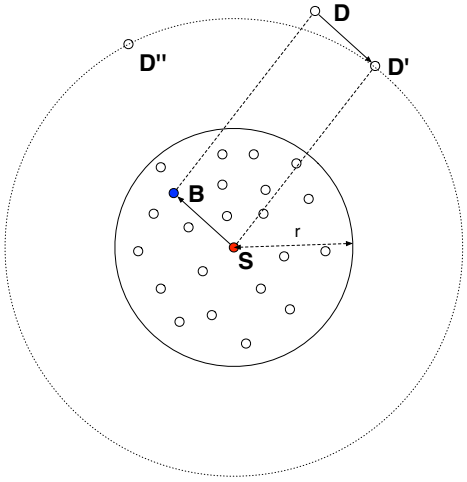



**Fig. 1.** One-hop neighborhood of $S$, translation and rotation of $D$

## 3 Mapping the simulated network to a hexagonal neighborhood

Our emulation approach improves upon the one proposed in our previous work [9], and differs from the other emulation approaches existing in the literature. We use realistic radio communications and aim to use the real radio links, without simulating them. This makes our approach fundamentally different from the other aforementioned emulation methods.

Our emulation framework uses a small wireless sensor network, up to 50 sensor nodes, to study the behavior of large scale wireless sensor networks. Instead of the random neighborhood structure described in 2.2, we deploy a set of sensors in a grid-like structure based on regular hexagons. Randomness and different densities of the 1-hop neighborhood are achieved at each step by using only a subset of the nodes in this regular structure.

Our 1-hop environment consists of 43 WSN430 sensor nodes placed in a hexagonal grid $P$ as shown in Fig. 2 (dotted part named *physical hexagonal grid* $P$). The origin $S$ is positioned at the center of this structure. At each emulation step, only node $S$ forwards the routed packet over a real radio link. The positioning of the sensor nodes following a hexagonal pattern ensures that in two successive steps, each node lying at the intersection of two successive neighborhoods, translates to one of the nodes of the real network grid, due to central and axial symmetry of the proposed structure.

Emulation using this regular hexagonal structure can be explained in the following way. Suppose that we have a network $H$ with nodes generated randomly or by following any desirable distribution. This network is composed of a large number $N$ of nodes to simulate, *e.g.* $N > 1\,000$, see Fig. 2. The source $U$ and destination $V$ are set for a routing task. We translate $U$ to the center $S$ of

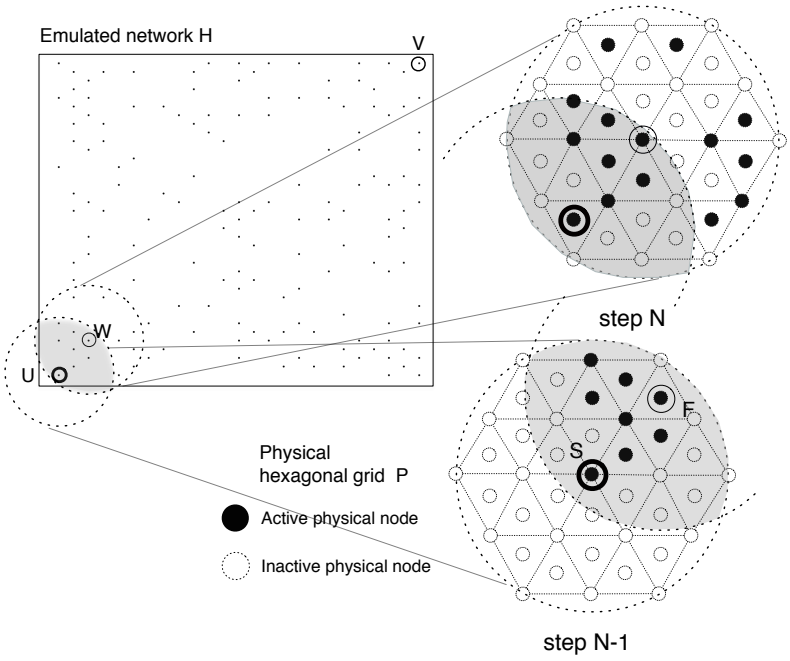



**Fig. 2.** Mapping the simulated neighborhood of the simulated source node to the real nodes surrounding the real source node of the physical network in two consecutive routing steps

our physical network $P$. Both networks have the same transmission radius — if necessary, additional scaling can be performed to achieve that. Destination $V$ is translated to virtual destination $D$. Each neighbor $W$ of the source node $U$ is translated to the virtual node $W'$ in the neighborhood of $S$. This virtual node is mapped to physical node $F$ by rounding its coordinates to the hexagonal grid, that is, it is mapped to its nearest physical node. The set of neighbors of $U$ is mapped to a set of neighbors of $S$, marked by filled discs in Fig. 2. This means that some neighbors of $S$ from $P$ are activated, while others are deactivated for the next routing step. We then use the physical links to activated nodes to select the forwarding node $F$ among them, whose origin is simulated node $W$ from $H$.

Then $W$ becomes the new source node. We translate $H$ further by vector $-\overrightarrow{UW}$, which then moves also the destination to a new position. This new neighborhood will map to a new set of activated nodes around $S$. Note that the set of previous and current source neighboring nodes overlap, and that this intersection results in a neighborhood structure that is preserved but translated, which is represented as a shaded region in Fig. 2. This procedure is repeated until we reach destination $D$ or the routing algorithm fails.

The main advantage of our method, even compared to the full size experimentation, is the ability to work with arbitrarily dense networks, by placing

the appropriate number of nodes in the 1-hop neighborhood. Furthermore, the source node $S$ can have a different neighborhood across successive steps, by using only a subset of the regular structure as candidate forwarders. Although highly regular, this structure is offering significant variety in node placement and distance to central node $S$. This approach allows to emulate a wide range of densities. However at too high densities, several simulated nodes could map to the same physical node, which poses the limit on this emulation.

The ultimate advantage of our method is its unlimited scalability. It can provide virtually unlimited network scenarios. We will obviously sacrifice some accuracy and fidelity, but are likely to gain much more insight compared to pure simulations. Our method can be applied on existing testbeds, from 50 to 1 000 nodes, to emulate the performance of even a million sensors network.

## 4 Experimental results

### 4.1 Experiments using the basic emulation setup

This section contains experimental results obtained using the basic emulation idea from previous work [9]. The experiment was done using a physical testbed consisting of 43 WSN430 sensor nodes [21]. The first phase of the experiment serves to gather statistics on link qualities, used directly afterwards for evaluating the routing algorithms. During this phase, each node transmits a batch of 128 messages, one node at a time, while the other nodes are measuring the number of successfully received messages from each node separately. Each node transmits 1 024 messages in 8 series. Statistics on link qualities from each separate node are gathered in the central node $S$ (referent node). This allows to compute $ETX(uv)$ (expected transmission count) for the whole 1-hop neighborhood link set and use it as part of the routing metric.

We calculate $ETX(uv)$ for each given link $uv$ as:

$$ETX(uv) = \frac{1}{p(uv) \cdot p(vu)}$$

where $p(uv)$ and $p(vu)$ are message reception probabilities based on measures over 1024 sent messages between nodes $u$ and $v$. For our case where nodes are static, the reception probability does not change significantly over time (standard deviation in the number of received messages in separate series was 4.61 %). This $ETX$ measure is used during the routing phase. We have evaluated the performance of three different geographical routing algorithms, XTC [19], GARE (greedy algorithm in real environment) and COP_GARE (cost over progress greedy algorithm in real environment) [9].

GARE is a localized greedy routing algorithm where the current node makes routing decision based on its position, the position of the destination, and its 1-hop neighbors. We calculate locally the Relative Neighborhood Graph (RNG) using $\frac{ETX(uv)}{|uv|}$ as a weight function for link $uv$. The longest edge in each triangle is not considered as a candidate for forwarding. Among RNG links, the one

which provides the largest progress towards the destination is selected. If there is no RNG edge with positive progress, the next node is chosen using the same criteria of minimal distance to destination, among the remaining edges.

COP_GARE is based on the COP algorithm [16] adjusted by using $ETX$ as a weight function. Among the neighbors with positive progress, we choose the one with the minimum ratio $\frac{ETX(uv)}{|ud|-|vd|}$, where progress $|ud|-|vd|$ is the difference between current distance from the destination $|ud|$ and possible distance $|vd|$.

Similarly to GARE, XTC [19] is based on the RNG structure, but it uses only $ETX$ as a weight function. This algorithm prefers the most reliable and at the same time the shortest links.

The success rate was 100 % for all algorithms due to the high density of the 1-hop neighborhood (the node degree of the source node was 41 at every step of the emulation) and the uniform placement of the sensor nodes which allowed the routing algorithms to have positive progress at every step.

All the routing algorithms demonstrated their expected behavior. XTC had the highest total number of hops and retransmissions, because it systematically selected short RNG edges. GARE algorithm shows overall improvement in energy consumption (smaller number of hops), using longer links with slightly worse $ETX$. COP_GARE outperformed GARE and XTC since it did not restrict to the RNG neighborhood subset. We present performance results of the three routing algorithms averaged over 100 runs in Table 1.

**Table 1.** Performance comparison with randomized graph orientation

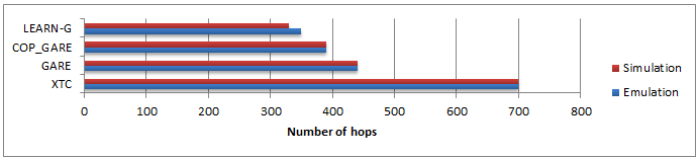
| Routing algorithm | Total No. of transm. | Total No. of hops |
|---|---|---|
| XTC | 6 196 | 5 820 |
| GARE | 4 948 | 3 103 |
| COP_GARE | 2 162 | 1 536 |

### 4.2 Emulation using a hexagonal grid neighborhood

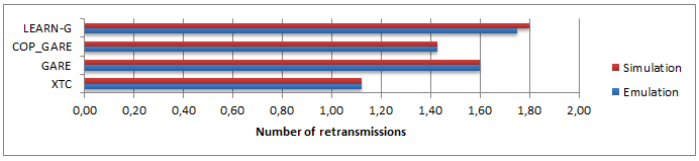
This new emulation setup is introduced to avoid using a rotation of the virtual destination, and to use only a subset of the available neighbors instead of the full set, as elaborated above.

The large scale simulated network was generated using an adapted version of the MIN-DPA algorithm [11]. MIN-DPA algorithm generated connected graphs with no crescent holes. In a nutshell, MIN-DPA calculates first an approximate transmission range $r$ such that the expected node degree is equal to $d$ (desired density). It then places $n$ nodes sequentially, in $n$ rounds. The $i$-th node is placed based on the positions of previous $i-1$ nodes. Approximate degrees $d_j$ of nodes already placed (based on $r$) are calculated. Proximity constraint is satisfied if node $i$ is not isolated from the previous nodes based on the approximate range

$r$ and it is no closer than $d_{min}$ to any of the previous nodes. The next node is placed in the neighborhood of node with minimal $d_j$ value.




(a) Average number of hops for routing algorithms




(b) Average number of retransmissions per hop

**Fig. 3.** Comparison of the results for emulation and simulation.

We performed two simulations, and compared same algorithms over the same network scenarios to show proof of concept of our approach and to emphasize its benefits. One simulation does not use any concept of emulation. It simply runs same algorithms on the full network. The results are labeled by "Simulation" in Figure Fig. 3. The second simulation follows our emulation setup. The actual emulation (physical testbed) was not carried. Instead, the whole process, as described here, was simulated. Thus we simulated the emulation process. Emulation was simulated using the placement of 43 sensor nodes in a hexagonal grid in a 1-hop neighborhood of radius $r = 5\,m$. The results are labeled with "Emulation" in Figure Fig. 3. Both "simulation" and "emulation" were carried based on the same physical layer models. In that sense, the comparison made shows the pure impact of our emulation, and results are more realistic then if the comparison were made with testbed simulation, where physical layer and its impact would be unpredictable, and comparison with "simulation" is difficult. Overall, Figure Fig. 3 shows small difference between "simulation" and "emulation" in all cases.

We proceed by evaluating the performances of different geographical routing algorithms based on $ETX$ measurements. As in 4.1, we implemented XTC, GARE, COP_GARE and additionally LEARN-G. LEARN-G is a variant of the LEARN algorithm [18] with an additional greedy step. In LEARN the source node $S$, aiming for destination $D$, chooses the neighbor $N$ in the restricted neighborhood, such that $\sphericalangle NDS \leq \frac{\pi}{3}$, that has lowest energy mileage, given as $Energy(SN)/|SN|$ where $Energy(SN)$ is the energy needed to send a message from $S$ to $N$. If there is no such neighbor LEARN-G switches to greedy that does not impose restrictions on the angle.

For each run we have 4 source nodes, situated in the 4 different corners of the network. The total number of nodes in this network is 1 012 with an average node degree of 10. Routing is performed across the diagonals to reach the destination situated in the opposite corner of our simulated network. We measured the total number of hops needed to reach the destination and the average number of retransmissions needed at each step.

We obtained a 100 % success rate for all the routing schemes (all of them are greedy with no face recovery phase) since all generated topologies were of sufficiently high density to allow greedy routing to make advance at each step.

The results are shown in Fig. 3 and are similar to the one presented in previous work [9]. XTC shows the largest number of hops, while LEARN-G has the smallest number of hops, since it is favoring longer links.

The performances of GARE and COP_GARE are situated between LEARN-G and XTC, since they are both using mid-sized edges. The biggest difference between simulation and emulation is for the case of LEARN-G, see Fig. 3(a): it has a slightly smaller number of hops per route for the case of simulation which can be explained by the fact that the algorithm was not constrained to the 1-hop neighborhood, as it was the case with emulation, but it could have also used neighbors outside the 1-hop neighborhood. The situation with the number of retransmissions is also similar, the smallest number of retransmissions is for XTC, GARE has the largest number, among the 3 algorithms used in [9], while COP_GARE performs in between having the lowest overall energy consumption. LEARN-G is using the longest links thus requiring the highest number of retransmissions. The difference in the number of retransmissions between emulation and simulation comes from the already mentioned difference in the links used.

## 5 Conclusion

Emulation improves the quality of results over simulation by taking into account more accurate environmental data or/and more realistic and complex models to represent sensor nodes or radio propagation. The main novelty in this paper is to imitate the behavior of a real large network deployment by using just small subset of the sensor nodes that would be required. With our previous work [9] we have given insight on the basics of the emulation and as well a proof of the feasibility of the real world implementation. In this paper we go a step further: we improve the emulation so it more closely resembles real world scenarios. Emulated networks can have an arbitrary number of nodes and a desired node distribution and density. Human effort and code complexity remain reasonable, still allowing large scale experimentation.

Our simulations, using the principle of small testbed of 43 WSN sensor nodes show that we can retrieve valuable and repeatable simulation results for a specific type of problems. We have created an environment that is using the same principle as emulation explained in this paper, but with higher consistency of

retrieved results. Future work will include experimental results with emulation on a real testbed. Mobility could also be included in our scenarios.

Our emulation approach can also be applied on already deployed large scale sensor testbeds, like GreenOrbs [8], the largest sensor network in the world which consists of 1 000+ deployed sensors. Would this network scale to 1 000 000+ nodes? What changes should be made to the CTP protocol to allow much larger deployments? The bottlenecks of existing large scale deployments could be studied and remedied using an emulation approach, as advocated here, on already large WSN, *e.g.* GreenOrbs would be an ideal platform.

This approach can be used for many other network layer studies, such as broadcasting, geocasting and multicasting for example.

## 6 Acknowledments

This work was partially supported by NSERC CRDPJ 386874 - 09 (Reliable and secure QoS routing and transport protocols for mobile ad hoc networks), NSERC Discovery grant of I. Stojmenovic, the French National Research Agency (ANR) project ARESA2 under contract ANR-09-VERS-017, ANR project IRIS under contract ANR-11-INFR-016, and the European Commission project CALIPSO under contract 288879.

## References

1. C. Burin Des Rosiers, G. Chelius, T. Ducrocq, E. Fleury, A. Fraboulet, A. Gallais, N. Mitton, T. Noël, and J. Vandaële. Using SensLAB as a First Class Scientic Tool for Large Scale Wireless Sensor Network Experiments. In *Networking 2011*, pages 241–253, Valencia, Spain, April 2011.
2. R. Canonico, P. Di Gennaro, V. Manetti, and Giorgio Ventre. Virtualization techniques in network emulation systems. In *Euro-Par Workshops*, pages 144–153, 2007.
3. G. Coulson, B. Porter, I. Chatzigiannakis, C. Koninis, S. Fischer, D. Pfisterer, D. Bimschas, T. Braun, P. Hurni, M. Anwander, G. Wagenknecht, S.P. Fekete, A. Kröller, and T. Baumgartner. Flexible experimentation in wireless sensor networks. *Commun. ACM*, 55(1):82–90, January 2012.
4. K. Fall. Network Emulation in the Vint/NS Simulator. In *Proceedings of ISCC'99*, pages 244–250, 1999.
5. Andreas Grau, Klaus Herrmann, and Kurt Rothermel. Efficient and scalable network emulation using adaptive virtual time. *International Conference on Computer Communications and Networks*, 0:1–6, 2009.
6. G. Judd and P. Steenkiste. Using emulation to understand and improve wireless networks and applications. In *NSDI*, 2005.
7. Q. Ke, D. A. Maltz, and D. B. Johnson. Emulation of multi-hop wireless ad hoc networks. In *Proceedings of the Seventh International Workshop on Mobile Multimedia Communications (MOMUC 2000), IEEE Communications Society*, oct 2000.

8. Yunhao Liu, Yuan He, Mo Li, Jiliang Wang, Kebin Liu, Lufeng Mo, Wei Dong, Zheng Yang, Min Xi, Jizhong Zhao, and Xiang-Yang Li. Does Wireless Sensor Network Scale? A Measurement Study on GreenOrbs. In *INFOCOM, 2011 Proceedings IEEE*, pages 873 –881, april 2011.
9. M. Lukic, B. Pavkovic, N. Mitton, and I. Stojmenovic. Greedy geographic routing algorithms in real environment. In *MSN*, pages 86–93, 2009.
10. S. Maier, D. Herrscher, and K. Rothermel. Experiences with node virtualization for scalable network emulation. *Computer Communications*, 30(5):943 – 956, 2007. Advances in Computer Communications Networks.
11. FA Onat, I. Stojmenovic, and H. Yanikomeroglu. Generating random graphs for the simulation of wireless ad hoc, actuator, sensor, and internet networks. *Pervasive and Mobile Computing*, 4(5):597–615, 2008.
12. S. Park, A. Savvides, and M. B. Srivastava. Sensorsim: A simulation framework for sensor networks. In *Proceedings of MSWiM*, August 2000.
13. Senslab. Very large scale open wireless sensor network testbed. http://www.senslab.info/.
14. A. Sobeih, J.C. Hou, K. Lu-Chuan, L.and H. Zhang Ning, WP Chen, HY Tyan, and H. Lim. J-sim: a simulation and emulation environment for wireless sensor networks. *Wireless Communications, IEEE*, 13(4):104 –119, aug. 2006.
15. I. Stojmenovic. Simulations in Wireless Sensor and Ad Hoc Networks: Matching and Advancing Models, Metrics, and Solutions. *Communications Magazine, IEEE*, 46(12):102 –107, december 2008.
16. Ivan Stojmenovic. Localized network layer protocols in wireless sensor networks based on optimizing cost over progress ratio. *IEEE Network*, 20(1):21–27, 2006.
17. J. Sventek, A. Maclean, R. McIlroy, and G. Milos. Xenotiny: Emulating wireless sensor networks on xen. Technical report, University of Glasgow, Department of Computing Science, 2008.
18. Yu Wang, Wen-Zhan Song, Weizhao Wang, Xiang-Yang Li, and Teresa A. Dahlberg. Learn: Localized energy aware restricted neighborhood routing for ad hoc networks. In *The 3rd Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks(IEEE SECON)*, 2006.
19. R. Wattenhofer and A. Zollinger. Xtc: A practical topology control algorithm for ad-hoc networks. In *IPDPS*, 2004.
20. WISEBED. Wireless Sensor Network Testbeds. http://www.wisebed.eu/.
21. WSN430. Kit developer's guide. http://perso.ens-lyon.fr/eric.fleury/Upload/wsn430-docbook/.