A Simulation Framework for Evaluating Complete Reprogramming Solutions in Wireless Sensor Networks

Michiel Horsman, Mihai Marin-Perianu, Pierre Jansen and Paul Havinga
University of Twente, The Netherlands
Email: m.horsman@student.utwente.nl, {m.marinperianu, p.g.jansen, p.j.m.havinga}@utwente.nl

***Abstract*—We propose a simulation framework developed in Simulink for analyzing the performance of code dissemination in wireless sensor networks. The complete solution relies on a three-layer network stack where the LMAC, FixTree and RMD protocols operate in conjunction. For performance evaluation, we use in our simulations the radio link quality model derived from previous field trials. In this way, we can study the impact of real network conditions (e.g. fluctuating link quality, changing neighborhood) on the higher layer protocols and thus verify our design choices in non-idealized circumstances.**

I. Introduction

In recent years, the interest in Wireless Sensor Networks (WSN) technology has expanded from environmental monitoring applications to different other areas, such as industrial processes [4], transport and logistics [9], healthcare, wellbeing [10], just to mention some. However, this widespread usage leads to much stricter requirements concerning the flexibility and reliability of WSN than initially conceived. In particular for industrial applications, the users need to have precise control and insight of what is happening in the network, and also to be able to tune or modify online the functionality of the deployed WSN. In this context, reliable wireless reprogramming of sensor nodes at scale represents a major problem. Traditional approaches based on flooding or viral, gossiping-like, code propagation [3] cannot be applied, since they provide a best-effort, "black-box" behavior to the users. Instead, a method that achieves a more robust topology control and a higher delivery ratio within guaranteed time bounds is needed.

In previous work, we proposed a reliable multicast data dissemination protocol (RMD) [8], which offers a good trade-off among reliability, scalability and energy efficiency. RMD was successfully used and demonstrated in the IST research project CoBIs [4]. Starting from these results, we propose a simulation framework for analyzing the performance of a complete WSN reprogramming solution. The simulation framework includes four major components:

1) A realistic radio link quality model derived from field trial results collected in previous experiments.
2) The data-link layer, represented by LMAC [13], a TDMA-based, lightweight medium access control protocol. LMAC is shown to be among the most energy efficient MAC protocols for WSN [5] and is appropriate for cross-layer interaction.
3) The topology control, implemented by a simple protocol for tree construction (FixTree), which represents the routing support of the dissemination layer.
4) The reliable dissemination layer, where the RMD protocol works in conjunction with FixTree and LMAC.

The implementation is carried out in Simulink, resulting in a modular, flexible framework that can be used to evaluate in detail the performance of the entire WSN stack.

The rest of this paper is organized as follows. In Section 2, we give a short overview of the related work. Section 3 describes the communication protocol stack. The simulator design is given in Section 4. Section 5 presents the method of developing the radio link quality model from field trial data. In Section 6, we present the detailed set of simulation results. Finally, Section 7 formulates the conclusions.

II. Related Work

Most simulators used for evaluating WSN communication protocols represent either adjusted versions of wired network simulators (e.g ns-2 and OMNeT++) or platform-specific tools (e.g. TOSSIM). Being an object-oriented simulator, ns-2 [11] is developed in C++, with an OTcl interpreter as a frontend. With respect to WSN simulation, ns-2 has to basic drawbacks [2]: the limited scalability due to the object-oriented design, and the lack of customization, as packet formats, energy models, MAC protocols, and the sensing hardware models differ from those found in most sensors. SensorSim [12] extends ns-2 in terms of power model, sensor channel model and interaction with the application layer. However, SensorSim still faces the scalability problem and is no longer being maintained. GloMoSim [14] simulation environment is based on the Parsec parallel discrete-event language. GloMoSim supports protocols for a purely wireless network and allows parallel simulations on multiple computers at the same time to gain speed. GloMoSim is no longer maintained. OMNeT++ [7] is an open-source simulation tool that provides a component architecture for models. Components (modules) are programmed in C++, then assembled into larger components and models using a high-level language (NED). TOSSIM[6] simulates WSNs running TinyOS. TOSSIM is essentially based on code that would also

978-1-4244-1653-0/08/$25.00

execute on real nodes. This makes the simulator very useful for development, but also creates a strict limitation on the generality of the simulated network.

The radio modeling has great impact on how close to reality WSN simulations are. Zhao et al. [16] study packet delivery performance in WSNs and compare three different environments. Zhou et al. [17] analyze the impact of radio irregularity, explaining the different ranges of the radio in specific directions. Furthermore, a radio irregularity model is created in [18]. Cerpa et al. [1] develop a method for characterizing statistically lossy links in WSNs based on real life measurements.

III. Solution Overview

For a thorough evaluation of WSN reprogramming, we use three protocols running at the data link, network and transport layers: LMAC, FixTree and RMD, respectively.

Data link layer - LMAC. LMAC [13] is a lightweight, energy-efficient medium access control protocol specifically designed for WSN. LMAC avoids collisions through scheduled access, where each node obtains periodically the right of using the medium for a fixed time interval, called time slot. Time slots can be reused over a 2-hop distance, thus ensuring the scalability of the protocol. LMAC provides several additional features that are beneficial for the higher network layers: one-hop neighborhood information, distance to gateway (for node-to-sink routing), and local acknowledgments (ACKs) and retransmissions.

Network layer - FixTree. FixTree represents a baseline for a minimal routing protocol. It is designed to create a spanning tree, over which the transport layer protocol disseminates the new data and code reliably to all receivers. The main requirements are simplicity and low footprint. Therefore, FixTree uses the standard hop-count tree formation approach. The source (or root) node floods an initial message in the network, and on the way back each node registers to a parent with smaller hop count. There are no further attempts to repair the tree if a link breaks. However, to extend the lifetime of the tree, FixTree uses link quality information in addition to hop count when choosing the parent node. As a result, the most stable links are preferred in building the tree.

Transport layer - RMD. At the transport layer, we use RMD [8], which is a cross-layer solution, utilizing MAC layer information about neighborhood and packet losses. In RMD, the source node starts to send the message divided in windows of packets. The packets are further pipelined down the tree by the intermediate nodes toward the leaves. Each packet is acknowledged by all the direct children using LMAC ACKs, which consume much less time and energy than individual ACK packets. If an intermediate node does not receive LMAC ACKs from some of its children, it retransmits the packet. This approach ensures that error detection and repairs are done locally. In order to have an indication of the correct reception of an entire window, we use window ACKs, which are standalone packets. Each intermediate node collects the window ACKs from all the direct children and forwards an aggregate window ACK upstream. In this way, the source node can safely advance to the next window of packets. By using this hybrid end-to-end – local acknowledgment mechanism, RMD ensures a trade-off between energy efficiency and delivery ratio.

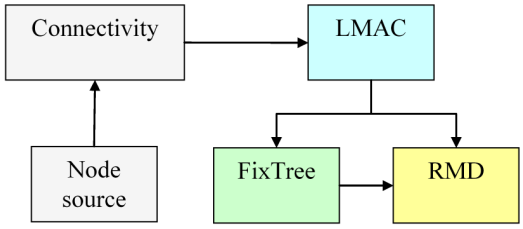


Fig. 1. Simulink model of the simulation framework.

IV. Simulink Model

We developed our simulation framework in Simulink, which is a widely used tool for modeling, simulating and analyzing dynamic systems. Being integrated with Matlab, Simulink offers direct access to all the analysis tools available in the Matlab environment. The primary interface is a graphical block diagramming tool and a customizable set of block libraries. A typical Simulink model consists of blocks organized hierarchically and connected by signals. The models are hierarchical, which facilitates both top-down and bottom-up approaches.

Figure 1 shows the Simulink model of the network stack described in the previous section. The functionality of the blocks is as follows. The *Node source* block generates the network topology (random or regular structure, see Section VI) and forwards it to the *Connectivity* block. The *Connectivity* block establishes the wireless links among the nodes and the link quality level, using data collected from a field trial (see Section V).

The *LMAC*, *FixTree* and *RMD* blocks implement the three communication protocols and interact with each other by exchanging matrices with relevant information. In the *LMAC* block, the nodes first compete for occupying a timeslot, then they go into LMAC normal operation mode (sleep state for most of the time, except the periodic control messages) and wait for messages to transmitted or received. After the initialization of LMAC is completed, the *FixTree* block activates the root node to build the spanning tree. Eventually, the *RMD* block uses the tree to disseminate a message to all the nodes in the network.

V. Link Quality Model

In our simulations, we are interested to study the behavior of the communication protocol stack under a realistic model of the wireless link quality. More specifically, we analyze the change of the link quality with time and distance. For this purpose, we use the results of a previous field trial [15], where 18 sensor nodes were deployed at our university campus. The nodes were scattered in different types of environments (in trees, within a parking lot and on building exterior walls). The trial lasted for 24 hours.

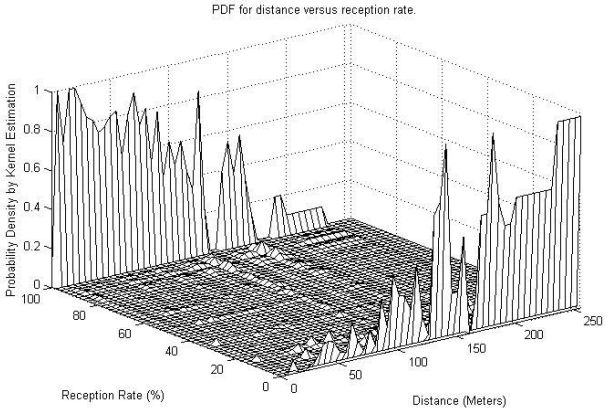


Fig. 2. PDF of link quality obtained from a node deployment.

In order to estimate the link quality, each node counted the LMAC control messages received from every neighbor over a period of 256 frames. It follows that a value of 255 indicates a very stable connection, while a 0 means no link at all. Even using this simple metric, the nodes had not enough storage capacity to keep the data locally, but transferred it periodically to a sink node connected to a PC. The resulting log file contains detailed information about the evolution of the link quality in time between each pair of nodes, i.e. for 153 distances (note that since this was a multihop deployment, there were pairs of nodes that never had a direct connection).

In order to derive a link quality model for our simulator, we have to extract the following information from the results of the trial:

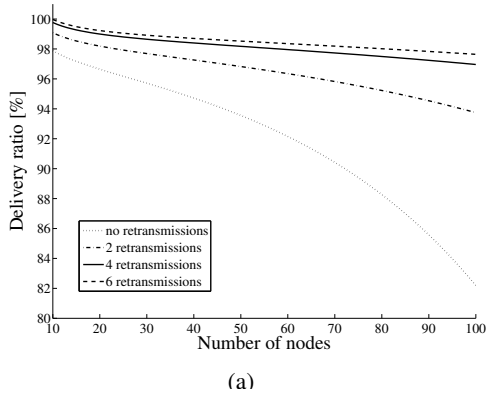
1) What is on average the initial link quality $q_0(d)$ between two nodes placed at any distance d from each other.
2) How does this link quality evolve in time, in other words, how do we predict $q_{t+1}(d)$ based on $q_t(d)$.

For the first problem, we use the procedure described in [1]. The probability that the link quality q_0 occurs at distance d is given by the amount of log entries found for the corresponding combination (d, q_0). The probability distribution function is represented in Figure 2. We can notice the clear tendency of the links to be either very good (for distances smaller than 100m) or very bad (for distances larger than 175m).

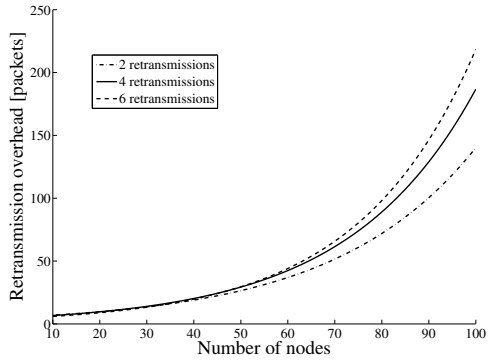
The second problem involves a more laborious procedure. For every distance d, we compute a stochastic matrix where the element at row i and column j gives the probability that the link quality $q_{t+1}(d)$ is j knowing that $q_t(d)$ was i. The following example helps clarifying this procedure. Suppose that for a certain distance d we have the following log fragment:

```
Time:     10  20  30  40  50  60  70  80  90 100 110 120 130
Quality: 255 255 255 255 245 255 255 255 255 255 145   0  10
```

We notice that link quality 255 is followed by 145 (1 occurrence), 245 (1 occurrence) and 255 (6 occurrences). Then, the stochastic matrix m^d corresponding to distance d has $m^d_{255,145} = 0.125$, $m^d_{255,245} = 0.125$ and $m^d_{255,255} = 0.75$. In the same way we can compute the values for link qualities 145, 245, etc.



(a)



(b)

Fig. 3. Establishing the optimal number of local retransmissions in RMD, as a trade-off between the delivery ratio (a) and communication overhead (b).

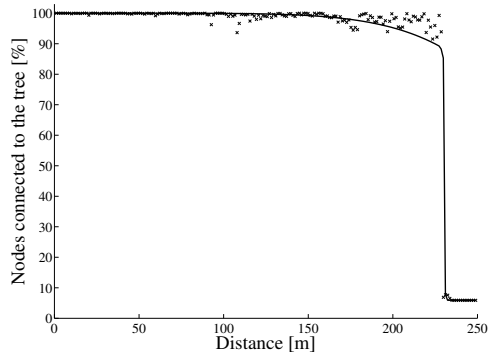
VI. Simulation Results

In this section, we study through simulations the influence of different network properties on the overall performance of the reliable code dissemination.

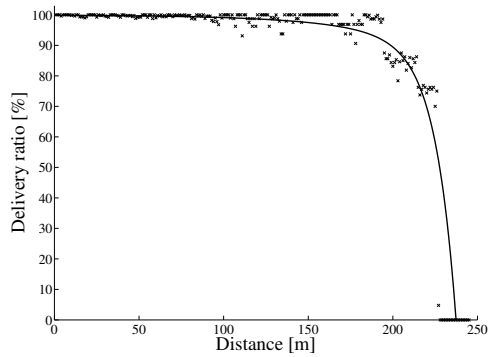
Experiment 1 - Local retransmissions in RMD. As a first step, we establish the optimal number of local retransmissions in RMD, as a trade-off between the delivery ratio and communication overhead. We consider the nodes randomly deployed and vary the number of retransmissions from 0 to 6. Figure 3(a) shows that up to 4 retransmissions there is a significant increase in the delivery ratio (up to 15%). However, from 4 to 6 retransmissions, the improvement is less than 1% and, additionally, the communication overhead grows exponentially, as plotted in Figure 3(b). Consequently, we will use 4 retransmissions as an optimal value throughout the rest of our experiments.

Experiment 2 - Distance. In this experiment, we are interested to find out the impact of increasing distances between nodes over the performance of the network. For this purpose, we arrange the nodes in a hexagonal grid, so that the distance between each pair of nodes is the same. We are interested in two performance metrics: (1) the percentage of node connected in the tree by FixTree and (2) the percentage of nodes that received and acknowledged correctly the message through RMD.

The results are plotted in Figure 4. We notice that FixTree manages to connect all the nodes in the tree for distances up

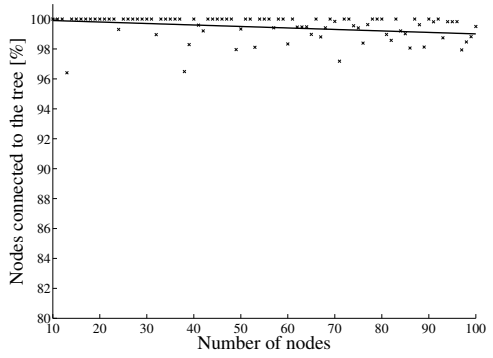


(a)

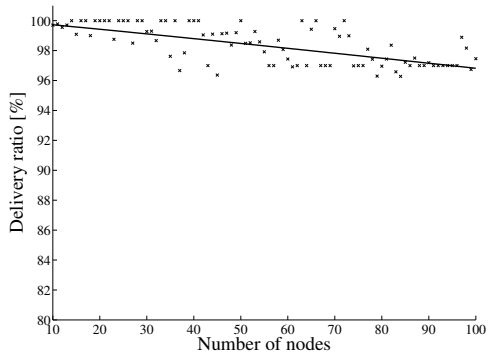


(b)

Fig. 4. The performance of FixTree (a) and RMD (b) when varying the distance between nodes (hexagonal grid deployment).



(a)



(b)

Fig. 5. The performance of FixTree (a) and RMD (b) when varying the network size and implicitly the hop count (star grid deployment).

to approximately 120m, and more than 90% of the nodes for distances up to approximately 175m. For distances larger than 230m the tree is practically impossible to build. Similarly, RMD achieves more than 99% delivery for distances up to 120m, and more than 90% for distances up to 175m. For distances larger than 175m, the performance of RMD drops significantly. From this experiment, we can conclude therefore that 120m is a "safe" distance threshold for optimal performance of the network stack.

Experiment 3 - Network size. In this experiment, we evaluate the performance in networks with larger hop counts. The nodes are deployed in a star-grid topology. The distance between two neighbor nodes on the vertical and horizontal axis is 75m, and between two neighbor nodes on the diagonal axis is 106m, respectively. Therefore, distance and density are not limiting factors in this experiment. By varying the number of nodes, we obtain average hop counts from 4 to 12 hops.

The results are plotted in Figure 5. We see that FixTree connects more than 99% of the nodes to the tree even for larger hop counts. Up to 50 nodes, RMD achieves a delivery ratio higher than 98.5%. For larger networks, the performance of RMD decreases linearly with the number of nodes and the hop count, getting to 96.8% for 100 nodes (12 hops).

Experiment 4 - Network density. In this experiment, we study the impact of the network density over the protocol performance. We consider the nodes deployed randomly in an area of constant size. Neither the number of hops, nor the distance are limiting factors. Figure 6 shows the performance of FixTree and RMD, respectively. Since LMAC is a TDMA protocol, we observe a clear limitation after exceeding the number of available time slots. The nodes that cannot occupy the wireless medium are left out the dissemination tree and, even if they can receive the messages, they cannot acknowledge the correct reception.

From experiments 3 and 4 we conclude that the deployment of a WSN must be carefully considered, in order to make the best trade-off between the number of hops and the network density. While the first factor affects linearly the delivery ratio, the latter poses a strict limitation to higher layer protocols.

Experiment 5 - Planning the deployment. As a final experiment, we analyze the situation of a carefully planned deployment, where the limiting factors identified so far are avoided. The results in Figure 7 show that for up to 50 nodes, almost 100% of the nodes are connected to the tree and receive and acknowledge the messages through RMD. For more than 50 nodes, the performance decreases very slowly. For 100 nodes, FixTree still connects 99.6% of the nodes to the tree and RMD delivers 98.8% of the messages correctly.

VII. Conclusions

We presented a simulation framework for analyzing realistically the performance of complete WSN reprogramming solutions. We used three example protocols at the data link, network and transport layer: LMAC, FixTree and RMD, re-

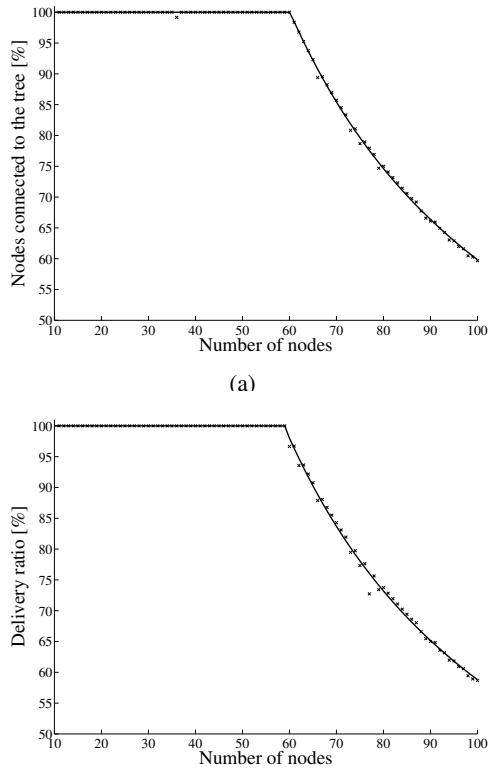


Fig. 6. The performance of FixTree (a) and RMD (b) when increasing the network density (random deployment).

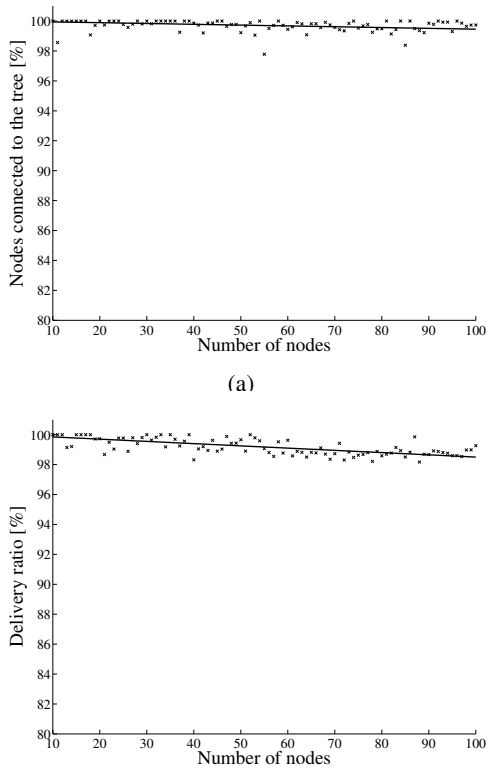


Fig. 7. The performance of FixTree (a) and RMD (b) in the case of a planned deployment.

spectively. Benefiting from the flexibility of Simulink and the rich processing capabilities of Matlab, we showed how a link quality model can be derived from real field data and incorporated in the simulator. The simulation experiments analyzed the impact of three important parameters (distance, network size and density) on the overall performance. The results indicated that a careful deployment can improve significantly the stability of the reprogramming solution, ensuring more than 98.8% average success rate for networks up to 100 nodes.

ACKNOWLEDGMENT

The authors would like to thank Supriyo Chatterjea and Yang Zhang for their support with the field trial data and Stefan Dulman for his advices on Simulink modeling.

REFERENCES

[1] A. Cerpa, J. Wong, L. Kuang, M. Potkonjak, and D. Estrin. Statistical model of lossy links in wireless sensor networks. In *IPSN*, 2005.
[2] David Curren. A survey of simulation in sensor networks.
[3] J. W. Hui and D. Culler. The dynamic behavior of a data dissemination protocol for network programming at scale. In *SenSys*, pages 81–94, 2004.
[4] Collaborative Business Items. http://www.cobis-online.de.
[5] K. Langendoen and G. Halkes. *Embedded Systems Handbook*, chapter Energy-Efficient Medium Access Control. CRC press, 2005.
[6] P. Levis, N. Lee, M. Welsh, and D. Culler. Tossim: accurate and scalable simulation of entire tinyos applications. In *SenSys*, pages 126–137, 2003.
[7] C. Mallanda, A. Suri, V. Kunchakarra, S.S. Iyengar, R. Kannan, A. Durresi, and S. Sastry. Simulating wireless sensor networks with omnet++. 2005.
[8] M. Marin-Perianu and P. Havinga. RMD: Reliable multicast data dissemination within groups of collaborating objects. In *LCN*, pages 656–663, 2006.
[9] R. S. Marin-Perianu, M. Marin-Perianu, P. Havinga, and J. Scholten. Movement-based group awareness with wireless sensor networks. In *Pervasive*, pages 298–315, 2007.
[10] Y. Nam, Z. Halm, Y. Chee, and K. Park. Development of remote diagnosis system integrating digital telemetry for medicine. In *Annual International Conference of the IEEE*, 1998.
[11] Network Simulator NS2. http://www.isi.edu/nsnam/ns.
[12] S. Park, A. Savvides, and M. Srivastava. Sensorsim: a simulation framework for sensor networks. In *MSWIM '00: Proceedings of the 3rd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, pages 104–111, 2000.
[13] L. van Hoesel, T. Nieberg, J. Wu, and P. Havinga. Prolonging the lifetime of wireless sensor networks by cross-layer interaction. *IEEE Wireless Communication Magazine*, 12 2004.
[14] X. Zeng, R. Bagrodia, and M. Gerla. Glomosim: a library for parallel simulation of large-scale wireless networks. In *PADS '98: Proceedings of the twelfth workshop on Parallel and distributed simulation*, pages 154–161, 1998.
[15] Y. Zhang, S. Chatterjea, and P. Havinga. Experiences with implementing a distributed and self-organizing scheduling algorithm for energy-efficient data gathering on a real-life sensor network platform. In *International Workshop on From Theory to Practice in Wireless Sensor Networks*, 2007.
[16] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *SenSys*, pages 1–13, 2003.
[17] G. Zhou, T. He, S. Krishnamurthy, and J. Stankovic. Impact of radio irregularity on wireless sensor networks. In *MobiSys*, 2004.
[18] G. Zhou, T. He, S. Krishnamurthy, and J.. Stankovic. Models and solutions for radio irregularity in wireless sensor networks. *ACM Trans. Sen. Netw.*, 2(2):221–262, 2006.