Interactive Tools for Learning Sensor Network Basics

Marie Paule Uwase[1], Jacques Tiberghien[2], Kris Steenhaut[3]

[1]Graduate student, Faculty of Applied Sciences and Engineering, VUB, Brussels, Belgium
Corresponding author email : marie.paule.uwase@vub.ac.be
[2]Professor, Faculty of Applied Sciences and Engineering, VUB, Brussels, Belgium
[3]Lecturer, Department Industrial Sciences, Erasmushogeschool, Brussels, Belgium.

ABSTRACT
Wireless sensor networks consist of autonomous intelligent sensor nodes, usually powered by battery, which can measure certain characteristics of their environment, such as temperature, pressure, moisture, acceleration, etc. These sensor nodes communicate by radio with their neighbours to forward data to a central collection points from where it is sent to some analysis centre via the Internet. The use of batteries and radio communications minimizes the cost of such a measurement system. Wireless sensor networks have a broad range of applications: climate monitoring, flood prevention, seismic monitoring, early detection of bush fires, etc. A major concern is the lifetime of batteries. As most of the energy is used for radio transmission, many research efforts have focused on the development of communication protocols that minimize power.

Many masters students are interested in wireless sensor networks as a research topic for their thesis, but don't know enough about it to make a rational choice. This led the authors to start implementing a user friendly multi-media e-learning course, explaining the essential aspects of sensor networks and their routing protocols. It has been observed that students, who have the opportunity to experiment freely with systems they need to understand, learn faster and more in depth than those who do not have such opportunities. For that reason, a very simple interactive simulator that shows the principles of routing has been included in the e-learning course. It does not replace in any sense the more sophisticated simulation environments that are available to researchers. On the contrary, it will be used to clarify basics of routing and to trace messages through real networks before confronting the students with the fully fledged simulation packages. The didactic value of the tool for helping students gaining insight in concepts underlying routing will be evaluated in four engineering institutes in three continents. In this paper the decision to build the didactic tool is motivated and the design issues are discussed, together with planned extensions.

Keywords: didactic tools; routing protocols; simulation; wireless sensor networks

1.0 INTRODUCTION

In the first chapter of his book "The HP Way" (Packard 1995) David Packard states that Prof. Fred Terman from Stanford University was instrumental in the creation of Silicon Valley by explaining in a simple and straightforward way the complexities of modern electronics to many generations of future engineers. The authors of this paper believe that indeed the in-depth understanding of basic principles, cleared from all fundamentally irrelevant but deemed important complex details, is the key to a successful engineering career. Working in a research group exploring protocols for Wireless Sensor Networks, they developed a simulator for routing protocols, intended for newcomers in the field. The objective was the creation of a didactic tool that will enable students to freely explore routing protocols.

2.0 WIRELESS SENSOR NETWORKS

Wireless sensor networks (WSN) consist of autonomous intelligent sensor nodes, usually powered by battery, which can measure certain characteristics of their environment, such as temperature, pressure, noise level, acceleration, etc. These sensor nodes communicate by radio with their neighbours to forward their measurements to a central data collection point from where all the data are typically sent to some processing and analysis centre via the Internet. WSNs have a broad range of applications such as climate monitoring, tropical flood prevention or early detection of bush fires (ITU-T, 2008). The use of batteries and radio communications minimizes the cost of setting up such distributed measurement systems, which makes this technology highly relevant for developing countries. A major concern with most sensor networks is the lifetime of batteries. As most of the energy drawn from the batteries is used for radio transmission, many research efforts during the last ten years have focused on the development of communication protocols that minimize power consumption while not jeopardizing the reliability of the data forwarding to the collection nodes. Most sensor networks use two different scenarios: one "collect protocol" to gather measurements and to forward them from node to node towards the data collecting node (the "sink") and one "disseminating protocol" able to send commands from an access point via multi hopping towards specific nodes. These protocols should be capable of adapting to the radio propagation conditions and to the state of charge of the batteries of the different nodes. The cost of a route is often expressed as the number of transmissions needed to relay a message from source to destination. Early algorithms just counted the number of hops but modern ones take into account the quality of the links by estimating the number of retransmissions needed for correct delivery of the message (Levis et al, 2008). A thorough insight in the basic concepts behind those algorithms is essential for those who want to research or even use WSNs.

3.0 SIMULATOR FOR WSN ROUTING PROTOCOLS

Many simulators for studying some aspects of WSN are available. Most are discrete event simulators specifically designed for simulating WSNs such as J-SIM (Tyan and Zhang, 2005) and JIST/SWANS (Barr, 2004) or general purpose discrete event simulators for computer networks such as OMNET++ (OMNET,2010) and NS-2 (NS2,2010). Particularly relevant in the author's environment, because used by researchers, is Cooja (Österlind et al, 2009), a simulator that comes with the Contiki operating system for WSNs and that allows sophisticated simulations of the radio links. All these simulators are intended for researchers who are already familiar with the field and not for newcomers who need to understand the basics before deciding whether they will invest intellectually in these topics. It might be possible to tailor some popular simulators for that purpose, but the authors preferred to develop a brand new, lightweight, didactic interactive simulation environment that could easily be integrated in a multi-media e-learning course. The aim of their simulator is to clarify the working of traditional routing protocols, to reveal how they behave when deployed in sensor networks and to demonstrate how derived protocols can perform better. They could not find equally simple and user friendly alternatives.

4.0 DESIGN OF THE SIMULATOR

4.1. Overall design

The first issue that needs to be tackled when designing didactic tools is how far real world problems can be simplified without becoming trivial and, more important, without giving the user an incorrect image of the real problem. In real WSNs, distribution and collect protocols often coexist, but with little direct interactions. These protocols have different goals and use different algorithms. Therefore it is logical to study them separately. As the collect protocol is likely to be most often used and therefore has a significant influence on the lifetime of the batteries, the first phase of the didactic simulations is restricted to collect protocols. Further restricting the protocols to networks with only one sink node has the advantage of limiting the

routing table in each node to a single entry and to allow for a simple graphical representation of the collect tree, by colouring the links that belong to that tree. The student can easily check, at each step of the routing protocol, whether every node has a link to the sink and, by simple additions along the path, verify that the routes are optimal. The aim is to build an interactive tool that can help motivated students to explore the behaviour of routing protocols in specific circumstances. The initial idea was to display the properties of all nodes and links on the graph during the routing process. It quickly became obvious that this was not a good idea: too much continuously changing information does not help the student in understanding precisely what is going on. It was decided to limit the displayed information to a minimum (in fact, just by changing the colour of the links involved in a particular route) but to give the user a large opportunity, between successive steps of the routing algorithm, to consult and possibly modify the attributes of all nodes and links, and to consult an overview screen that gives global statistics such as total number of nodes with exhausted batteries or the global usage of electrical power for routing purposes.

The Distance Vector (DV) routing algorithm is used to introduce the simplifications that were considered acceptable. Thereafter, the Ad-hoc On-Demand Distance Vector Routing (AODV) is simulated to get closer to protocols that are applicable in real deployments of WSNs. These protocols (described respectively in sections 5.2 and 5.3) were selected because the collect protocols in two popular operating systems for WSNs, TinyOS (TinyOS,2010) and Contiki (Dunkels,2004) are directly inspired by them.

4.2. Simulation of the state of the battery

An essential part of any routing algorithm in sensor networks is disappearing nodes, because their battery gets exhausted or connectivity fails. For that reason an integer attribute has been included in the attributes of every node (except the sink) to simulate the state of the battery. Whenever that variable reaches a predefined value, the node is considered as dead and all the links to it are disabled. All links in the simulator have a “cost” or a “distance” attribute that is set by the user when defining the network and that can be modified between routing steps just by double-clicking the corresponding link and entering a new cost value. The cost of a link can be defined arbitrarily, but the authors consider it as a measure of the number of attempts to be made to transfer a message and therefore the battery attribute of a node is decreased by the cost of a link each time a routing message is transmitted over that link (this rule is not applied to the sink as it is considered to be externally powered). For broadcasts, a fixed amount of power is subtracted from the battery of the broadcasting node. Reducing the remaining power only for transmitted messages is a serious simplification, as it is well known that receiving a message also uses a lot of energy (Borms et al., 2009). Although it would be rather simple to include the receive energy budget in the routing algorithm, it has been left out because it made node lifetime more difficult to follow during simulation runs and does not contribute significantly to understanding basics of the routing algorithms.

5.0 IMPLEMENTATION.

The simulator is written in Java since the students should be familiar with that language and it provides outstanding facilities for developing graphical user interfaces. From a portability point of view it would be desirable to implement it as an applet, because this would allow students to run it inside a web browser on any computer with any operating system, without having to do any installation efforts. However, for security reasons, access to system resources such as files and IO ports is severely restricted for applets. This makes it difficult to save network topologies for running several simulations on the same network. Therefore the software has been developed as a Java application, but the authors believe that when HTML5 will be fully supported by the browsers, applets will be the better choice. Making the change should not require big programming efforts.

5.1 The graphical user interface.

Central in any modern didactic tool should be a good graphical user interface. It should allow building interactively a representation of a WSN, define the properties of the components of that WSN and display graphically the results of the routing algorithms. To avoid reinventing from scratch software to build and display graphs, an open source package called "Graphe" that is available on the web (Pignede,2010) is used. This package is the result of a student project with very different objectives, but it provides "clean" data structures for representing graphs and assigning attributes to the nodes and the links, as well as interactive graphic methods to build the graph and setting the values of the attributes. In addition it provides methods to save graphs on file, to restore these graphs and even to create JPEG pictures of the graph. The software proved to be reasonably bug-free and quite easily adaptable for this project. The user interface of the simulator is shown in fig. 1. Normal nodes are represented by a circle, the sink node (00 in the example) by a double circle and the cost of each link by a figure surrounded by a square box in the middle of the link.

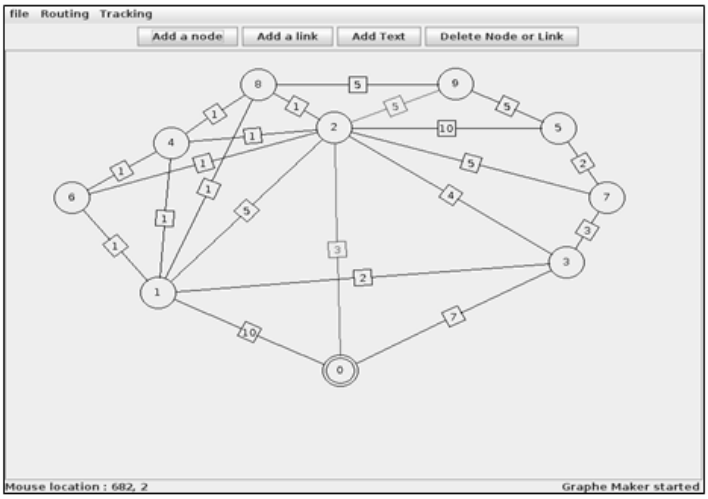

Fig.1. The graphical user interface of the simulator.

5.2 Distance Vector Routing.

Distance Vector Routing was the first routing algorithm used on a large scale on the Internet. It is a simple distributed algorithm that requires little communication between nodes. So, it is not surprising that this algorithm has inspired many designers of routing protocols for WSNs. For a clear explanation of the algorithm, we refer the reader to Andrew Tanenbaum's book Computer Networks (Tanenbaum, 2003). It can be summarized as follows: periodically each node asks all its neighbours what they believe their cost for reaching the sink is. From the received answers and the cost of the links to the different neighbours each node can compute its best known cost for reaching the sink. This process converges towards a state where each node knows what its optimal cost is and via which neighbour that optimal route starts. An alternative way of implementing the DV algorithm in WSNs consists in having the nodes broadcasting periodically their best known cost for reaching the sink and having their neighbours updating accordingly their own cost for reaching the sink attribute.

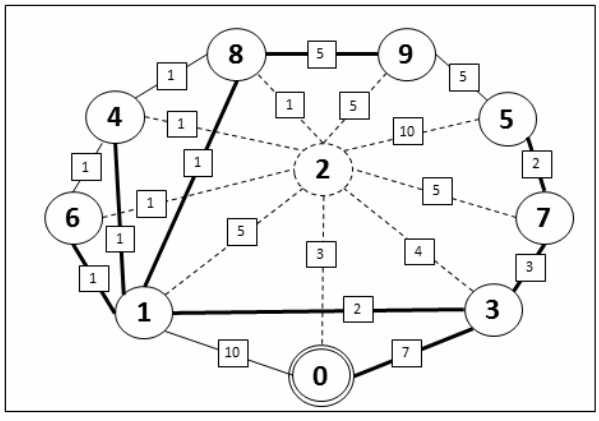

Fig.2. Distance vector routing after death of node 2

If these broadcast messages are acknowledged, the power consumption is the same as for the original implementation. Without acknowledgements, serious power saving are done, but the cost of the links cannot be measured by the routing process alone. For evaluating the state of the battery in this simplified didactic simulator, the original version of the algorithm has been used: for asking its neighbours, a node consumes the power corresponding to a broadcast and for answering such a question, it is the cost of the link that is taken from the battery.

The simulated routing algorithm is implemented as a for-loop that scans all nodes. For each node, first the state of the battery is checked and if exhausted, the node is declared dead with all links attached to it disabled. For nodes that are alive, all links are scanned to find the neighbour nodes that can be used to reach the sink. Each node has an attribute that gives the lowest known cost from the node to the sink (this attribute is initialized to infinity) and each link has an attribute that gives the cost of sending a message over that link. This cost is defined by the user when the graph is created and can be modified between routing steps. For each neighbour node, its cost for reaching the sink is added to the cost of the link used to reach the neighbour. This sum gives the cost of reaching the sink via this specific neighbour. When all links have been scanned, the routing attributes of the node are updated by retaining the lowest cost for reaching the sink and the corresponding link. When all nodes have been scanned once, the results are shown by colouring the dead nodes and links in gray and the selected links in red. The scanning cycle can be repeated step by step, or continuously at an adjustable pace. Fig.2 shows the results of a DV routing simulation with a full tree of optimal routes to the sink and one dead node due to battery exhaustion by the routing process.

In a real network, each node should, at regular intervals, ask all its neighbours what their distance to the sink is and update accordingly its own routing table. The actions of the different nodes are not synchronized, and as a consequence, the updates of routing tables are done in an almost random order. This randomness makes the understanding of the successive steps of the routing algorithm difficult for beginners while it does not really contribute to the understanding of the algorithm. In the first version of the simulator, the nodes polled their neighbours in a fixed order, determined by the index of the node in the array list of nodes. By running many examples with the simulator, it can be observed that this fixed order, linked with the way the network topology had been entered in the simulator, did sometimes create systematic routing transients of no didactic value. To avoid this, a randomizing hashing function has been inserted in the loops that determine the order of processing the nodes and links, giving a more random nature to the route finding. The randomization avoids systematic transient artefacts but makes predicting routing decisions much more difficult. Finally, it was decided to make the randomization optional through "Random" and "Sequential" entries in the routing menu.

5.3 Ad-hoc On Demand Distance Vector Routing.

The AODV algorithm was introduced as a simple approach to routing for mobile devices connected to the Internet (Perkins and Royer,1999). Instead of updating continuously routing tables in all nodes, this algorithm waits until a message needs to be transmitted to search for a suitable route to the destination of the message. This "just in time" approach potentially can minimize the traffic needed for rout-routing, and, therefore, is attractive for wireless sensor networks. An accurate explanation of the algorithm by its inventors can be found on the

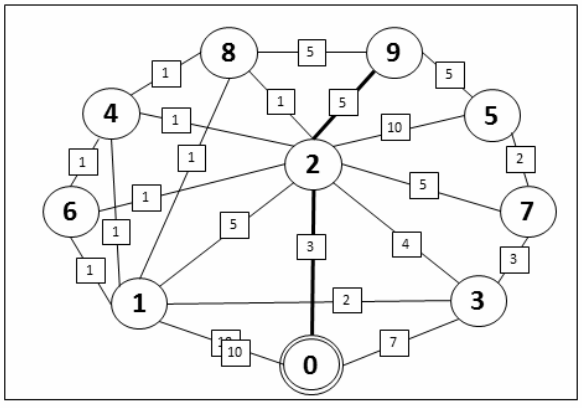

Fig.3. Optimal route found by AODV for node 9.

web (Das et al., 2010).

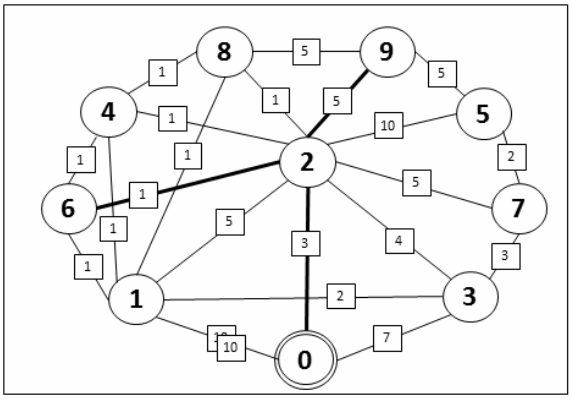

Fig.4. Additional route found by AODV for node 6

In AODV, there is no such concept as a periodic routing cycle: at the very moment that a node needs to send a message to the sink, it tries to find a good route for doing so, possibly reusing part of a route discovered previously by another node. Initially this process was simulated by selecting at random times a random node that initiates the routing process. This did not give convincing results from a didactic point of view as either the process was very dull or routes were changed before they were understood by the observer. Instead, the user is allowed to manually select the node that initiates a routing request by just double clicking it while the AODV routing process is active. To find the best route from an arbitrary node to the sink, a recursive backtracking method is used: all possible routes to the sink or to any other node that already has a route to the sink are compared and the best one is retained. To avoid exploring loops, each recursive search is given a unique, monotonously increasing, identifier that is stored as an attribute in all nodes that have been visited during the search. In the present version of the AODV simulation software, once a node initiates the routing process, the optimal route is immediately selected, while in the real AODV protocol, as soon a route is discovered, this route is used for data transmission, while the discovery process continues to search better routes. This stepwise route refinement is planned for a future version of the simulator, but its implementation is difficult in the present structure of the Graphe program.

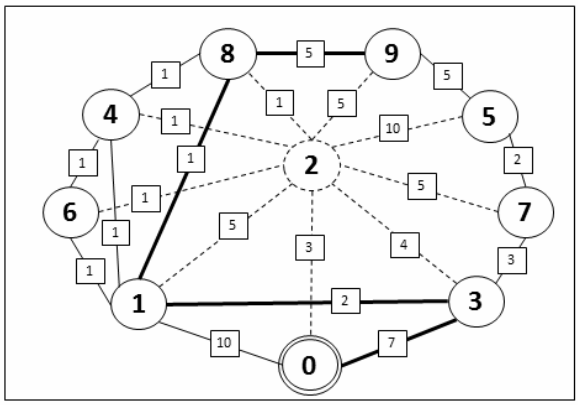

Fig.5. New route found for node 9 after death of node 2.

With AODV, when a node becomes inactive (due to battery exhaustion, for instance), a Route Error message is sent upstream to force all nodes using a route crossing the inactive node to give up that route. When this occurs, the affected routes are erased in the simulation and the user can request a new search by selecting an affected node. Erasing the affected routes is also done by means of a recursive method: it starts from the dying node and searches for all neighbours that had an optimal route via that node and resets the attributes relative to that route. Then, recursively, it operates similarly from all the neighbours, until there are no more nodes relying upon the failing route in the network. To allow for power related comparisons with the DV algorithm, similar battery accounting rules have been adopted: nodes use the broadcast mode to ask neighbours if they "know" how to reach the sink. When a node knowing a route to the sink has been found, the messages reporting that event back to the node that initiated the search are considered point to point. Messages reporting that a route has died are also considered as point to point.

Fig.3 shows the optimal route found for node 9, fig.4 shows how part of this route is reused when node 1 requests a route and finally, fig.5 illustrates what happens to these routes when the battery of node 2 gets exhausted.

6.0 CONCLUSION AND FUTURE WORK.

A correctly working simulator has been made. It has already allowed the authors, while playing with it, to become aware of many particularities of the routing protocols they thought they knew pretty well. The next step is to expose students to it and to find out, preferably in a quantitative way, if the tool helps them effectively for learning routing basics. The source code of the simulator, written in Java, can be obtained from the authors upon e-mail request.

7.0 REFERENCES.

Barr,R. (2004), SWANS– Scalable Wireless Ad hoc Network Simulator: User Guide, obtained from http://jist.ece.cornell.edu/docs/040319-swans-user.pdf

Borms,J., Steenhaut,K., Lemmens,B., Nowé,A. (2009), Power Aware Fulfilment of Latency Requirements by Exploiting Heterogeneity in Wireless Sensor and Actuator networks, *12th EuroMicro Conference on Digital System Design*, pp: 597 - 600, IEEE Computer society, ISBN-ISSN: 978-0-7695-3782-5

Das,S., Perkins,C., Royer,E., Marina,M. (2010), Ad Hoc On Demand Distance Vector. Obtained from http://moment.cs.ucsb.edu/AODV/aodv.html

Dunkels,A., Grönvall,B., Voigt,T. (2004): Contiki - a Lightweight and Flexible Operating System for Tiny Networked Sensors, Local Computer Networks, 29th Annual IEEE International Conference, page 455-462

ITU-T (2008), Technology Watch Report #4:Ubiquitous Sensor Networks.

Levis, P., Brewer, E., Culler, D., Gay, D., Madden,S., Patel,N,. Polastre,J., Shenker,S., Szewczyk, R., Woo,A., (2008), The Emergence of a Networking Primitive in Wireless Sensor Networks, Communications of the ACM, Volume 51, Number 7.

NS2 (2010), Main Page, obtained from http://nsnam.isi.edu/nsnam/index.php/User_information

OMNET++ version 4.0 (2010): user manual, obtained from http://www.omnetpp.org/doc/manual/usman.html

Österlind,F., Dunkels,A., Eriksson,J., Finne,N., Voigt,T.,(2009): Cross-Level Sensor Network Simulation with COOJA, obtained from http://www.sics.se/~fros/osterlind06crosslevel.pdf

Packard,D.(1995), The HP Way, Harper Collins, ISBN 0-88730-817-1.

Perkins,C., Royer,E. (1999), Ad-hoc On-Demand Distance Vector Routing, Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, pp. 90-100.

Pignede,PE.(2010), Graphe Maker, obtained from http://licence.kaillou.net/licence

Tanenbaum, A.(2003), Computer Networks, 4th ed. Pearson Education. ISBN 0-13-038488-7

TinyOS Extension Proposal (TEP) 119 (2010), Collection, obtained from http://www.tinyos.net/tinyos-2.x/doc/html/tep119.html

Tyan,L., H. Zhang,H.(2005), J-Sim: A simulation and emulation environment for wireless sensor networks., Proc. Annual Simulation Symposium (ANSS 2005).