

Experiences with Implementing a Distributed and Self-Organizing Scheduling Algorithm for Energy-Efficient Data Gathering on a Real-Life Sensor Network Platform

Yang Zhang, Supriyo Chatterjea, Paul Havinga

Department of Computer Science,
University of Twente, P.O.Box 217 7500AE,
Enschede, The Netherlands

Email: {zhangy,supriyo,havinga}@cs.utwente.nl

Abstract

We report our experiences with implementing a distributed and self-organizing scheduling algorithm designed for energy-efficient data gathering on a 25-node multihop wireless sensor network (WSN). The algorithm takes advantage of spatial correlations that exist in readings of adjacent sensor nodes and utilizes cross-layer information from the underlying MAC layer to minimize message transmissions. We describe how we modify our experiments in order to meet the assumptions made in the earlier theoretical analysis of the algorithm. The implementation results which are virtually identical to the preliminary simulation results, show that the algorithm achieves up to 80% energy savings when compared to conventional raw data collection.

1. Introduction

Scientists at the Australian Institute of Marine Science (AIMS) [1] are engaged in setting up a large-scale wireless sensor network to monitor various environment parameters on the Great Barrier Reef (GBR). Our proposed Distributed and self-Organizing Scheduling Algorithm (DOSA) [6] that will be deployed on the GBR allows certain nodes in the network to aggregate raw data in an energy-efficient manner by taking advantage of spatial correlations that exist in sensor readings of adjacent nodes. Moreover, the scheduling algorithm enables nodes to autonomously reassign schedules when the network topology changes due to failing or newly added nodes detected by cross-layer information provided by the underlying MAC layer.

The preliminary simulation framework [6] built in Matlab [2] simulates a large-scale sensor network and shows that DOSA results in an 80% reduction in message transmissions compared to conventional raw data collection.

Here we describe how we have implemented DOSA on a 25-node multihop WSN using cross-layer information from the underlying Lightweight Medium Access Control (LMAC) [5] protocol. We highlight some of the steps we have taken to ensure that the assumptions made in theory can also be made in real-life. We also describe the similarities and differences between the simulation and implementation. Through the implementation of DOSA, we achieve virtually identical results to the simulation results.

The rest of this paper is organized as follows. The details of DOSA and the simulation framework are introduced in Section 2. The implementation framework, hardware and software used in the implementation are described in Section 3. The specific difficulties faced and lessons learned in the implementation are described in Section 4. The similarities between the implementation results and simulation results of DOSA are shown in Section 5, and concluding remarks are made in Section 6.

2. Theory

2.1. DOSA Motivation and Overview

Conventional raw data collection in WSNs results in a very large number of data transmissions within the network, which results in rapid depletion of the batteries. Moreover, the limited bandwidth of nodes leads to dropped packets and thus decreases the quality of data. Additionally, network topology may change as sensor nodes are prone to failure or new nodes may be added to the network. Currently, although a variety of energy-efficient data acquisition techniques have been proposed for WSNs, they have a few drawbacks, e.g., Direct Diffusion [8] and TinyDB [9] are unable to collect raw data efficiently; BBQ [10] is unable to deal with the network topology changes; PAQ [11]

does not utilize any cross-layer information to improve efficiency.

DOSA uses a distributed graph coloring technique [3] to assign colors or schedules in order to decide when a particular node needs to act as a correlating node. The correlating node delivers correlation information representing the sensor readings of its adjacent neighbors to the sink. Thus correlating and non-correlating nodes are represented at the sink through real and estimated readings respectively. The sink then estimates the readings of the adjacent neighbors of the correlating node, by combining the current readings of the correlating node with the previously received correlation information.

The algorithm makes use of the high degree of spatial correlations that exist among the sensor readings of the adjacent nodes in a densely deployed network. The distributed property enables nodes to autonomously choose schedules based on locally available information. Every color owned by a node represents a time period during which the node will act as a correlating node. Moreover, the self-organizing property enables nodes to reassign schedules within a finite time if the network topology changes due to the failure or addition of nodes detected by cross-layer information provided by the underlying MAC layer. DOSA's constraints ensure that adjacent nodes do not act as correlating nodes simultaneously and that every non-correlating node is always within one-hop of at least one correlating node at any point in time. Thus every node will always be represented by either a real or estimated reading [6].

2.2. Simulation Framework

The simulation in [6] was implemented in Matlab and was based on 100 randomly placed nodes, each with a static topology. The simulation results investigated the following: (i) the number of nodes that actually transmit data (i.e., act as correlating nodes) as opposed to raw data collection (when all nodes transmit data) when the average node degree is varied from 5 to 11 by changing different transmission ranges for the nodes. (ii) time taken and number of messages transmitted to reassign schedules when a node is removed. (iii) time taken and number of messages transmitted to reassign schedules when a node is added randomly to the network. The simulation results show energy savings of up to 80% when compared to collecting raw data.

3. Experimental Setup

3.1. Implementation Framework

Figure 1 shows the overall organization of the deployment of 25 nodes distributed in an area of $2 \times 2 m^2$, including one sink node. Although all nodes are within transmis-

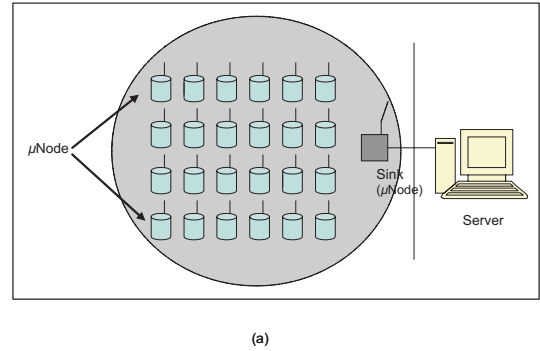


Figure 1. The implementation framework

sion range of each other, we simulate a multihop network in order to evaluate DOSA. The sink forwards results collected from the sensor nodes through a serial port to the server, which then analyzes the data to evaluate DOSA. As with the simulation, the implementation results investigated the following: (i) the number of nodes that actually transmit data when the average node degree is varied from 5 to 9. This operation is carried out for 50 randomly generated network topologies in each average connectivity. (ii) time taken and number of messages transmitted to reassign schedules when a node is removed. This operation is carried out for 50 randomly generated network topologies with an average connectivity of 8. (iii) time taken and number of messages transmitted to reassign schedules when a node is added. This operation is also carried out for 50 randomly generated network topologies with an average connectivity of 8.

3.2. Hardware and Software

1. *Ambient μNodes*: Figure 2 shows one of the 25 nodes used in the implementation framework. The Ambient 2.0 μNode contains an ultra low-power radio transceiver, a 4.6 MHz Texas Instruments MSP430 microcontroller featuring 10 KB of Random Access Memory (RAM), and 48 KB of programmable flash memory. This 16-bit processor features extremely low active and sleep current consumption that allow the μNode to run for years on two AA batteries. The board contains a versatile digital and analog I/O interface, which connects to multiple sensors and actuators. A JTAG interface is used to programme the CPU.
2. *AmbientRT operating system [4]*: AmbientRT is a Real-Time Operating System (RTOS) for embedded devices with very limited memory, processing power and energy resources. Despite these resource limitations, AmbientRT has very powerful features like (soft) real-time scheduling, dynamic memory alloca-

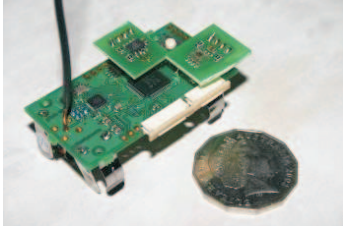


Figure 2. The Ambient μ Node

tion, online reconfigurability, and support for a data driven architecture.

3. *LMAC*: A TDMA-based MAC which uses a distributed algorithm to assign available time slots to nodes. A time slot consists of two sections, the Control Message (CM) and the Data Message (DM). A node can only transmit in its own slot once every frame. Nodes autonomously reassign slots when a topology change occurs. LMAC requires every node to maintain its local neighborhood information. LMAC effectively avoids the competing for medium and saves energy by minimizing collision and overhearing. Moreover, LMAC helps minimize the number of transceiver switches and reduces the complexity of the implementation.
4. *DOSA*: The scheduling algorithm that makes use of neighborhood information from LMAC for schedule assignment and adapting to topology changes. It includes three main operations, normal initialization, coping with a dead node and coping with a new node. While the footprints of LMAC and the AmbientRT is 2782 bytes, DOSA only takes up 873 bytes.
5. *Data collection layer*: It allows data to be collected from the entire network to evaluate the performance of DOSA. The whole implementation structure is shown in Figure 3.

4. Difficulties and Lessons

4.1. Creating a Stable Multihop Network

One of the most important assumptions in DOSA is that the network is static. Thus in a simulation environment, when two nodes are within transmission range of each other, we assume that they remain neighbors during the entire lifetime of the network. However, based on our initial deployment of 22 nodes in a forest in our university campus, we realized that the quality of radio communication is a very dynamic attribute. Figure 4 shows the link quality between two neighboring nodes. The link deteriorated around the 100-minute mark for around 1.5 hours due to heavy rainfall.

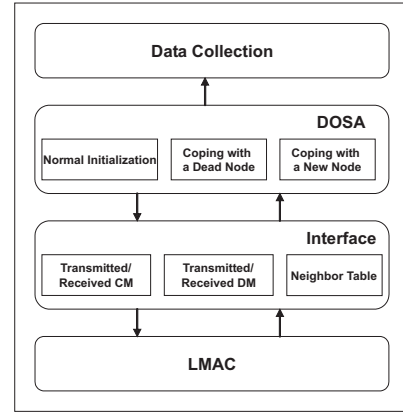


Figure 3. The implementation structure

This degradation of link quality results in frequent topology changes. This will cause DOSA to continuously reassign schedules to the affected nodes due to "missing" or "new" nodes, thus causing excessive overhead in a forest for testing DOSA. Thus while cross-layer optimization does allow a significant reduction in message transmissions [6], it also makes it prone to excessive overhead.

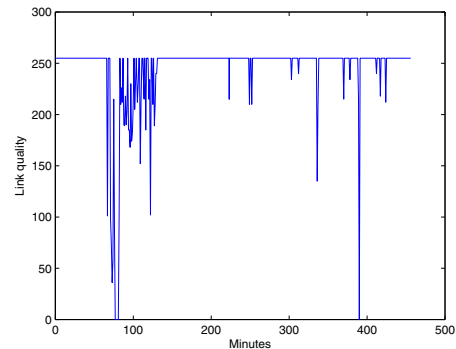


Figure 4. Impact of link quality between nodes in outdoor conditions

As the main objective was to study the energy-saving and self-stabilizing properties of DOSA, we carried out tests on an indoor network of 25 nodes which were all within range of each other, thus forming a complete graph. A large number of 25-node topologies are first randomly generated. These topologies are broadcast individually to the entire network. Each node then creates a virtual set of neighbors based on the received connectivity information, by ignoring the CM and DM transmitted from "nonneighbors". This results in a stable multihop network that is suitable for testing DOSA.

4.2. Managing the Removal and Addition of Nodes

In a simulation environment, it is very simple to add or remove nodes. A real deployment however, results in additional complexities especially because we cannot manually add or remove nodes. The process needs to be automated. However, a "removed node" cannot be completely deactivated as it should be able to receive a message from the sink requesting it to "add" itself back to the network. Thus, in order to remove a node, the sink sends a message to the chosen node. The node then causes its adjacent neighbors to execute DOSA on coping with a dead node (i.e., the node itself) but continues to execute LMAC. Once the neighbors of the "removed" nodes have reorganized their schedules and all the required data has been collected, the sink informs the chosen node to choose a new non-colliding LMAC slot, "rejoin" the network and choose a new DOSA schedule. Also, once the operation of DOSA about coping with a new node (i.e., the "rejoin" node) is finished and all the required data has been collected, the sink informs another chosen node to repeat the above process of the removal and addition of nodes again, until all of nodes are informed in the network.

4.3. Alleviating Memory Overflows

Recall that all randomly generated topologies are broadcast individually to the entire network in a simulation environment. A new topology will be broadcast automatically to the network when all operations of testing DOSA in a topology have been finished. However, in an implementation environment, the sink does not have enough memory to store all topologies, and also it is impractical to load every topology to the sink manually, e.g., using HyperTerminal. Thus we have created an application that can store all topologies and automatically send every topology to the sink, which then broadcasts it to the network.

Another problem may occur when the implementation results are forwarded to the sink in a real-life multihop network. The simulation results generated by nodes are collected completely in a distributed manner. However, in a implementation environment, an intermediate node does not have enough memory to store all of implementation results generated from other nodes and itself, so that some results may be lost before arrive at the sink. This will influence the effect of evaluating DOSA. Thus we allow that the implementation results generated by nodes are delivered directly to the sink instead of using the multihop network.

4.4. Solving Time Synchronization

We assume that each node in the network synchronizes its own clock with the sink in a simulation environment all

the time. However, in a real deployment, when they are initially powered on, it is possible that not all of nodes could successfully synchronize their clocks with the sink due to hardware limitations of the nodes. This prevents the unsynchronized nodes from receiving correct topology information. However, using existing algorithms that solve time synchronization in WSNs would increase the complexity of implementation. Thus we simply ensure that the sink maintains LMAC neighbor information of all other nodes in the network. The sink sends the topology information to the entire network individually, and then waits for an acknowledgement message generated by every node. If the sink does not receive the acknowledgement message from a node, we know that the node is not yet synchronized. Then the node will be powered on again, and the process is repeated until the node is synchronized.

5. Experimental Results

The implementation results are collected from the entire network to evaluate the performance of DOSA. After normal initialization, each node delivers its own acquired DOSA colors, which follow the same format as the LMAC colors transmitted to the sink. DOSA's color information indicates how many nodes in the network act as correlating nodes in a certain time slot of every frame. Only the correlating nodes deliver correlation information representing the sensor readings of their adjacent neighbors to the sink. In conventional raw data collection, every node would be required to transmit individual readings.

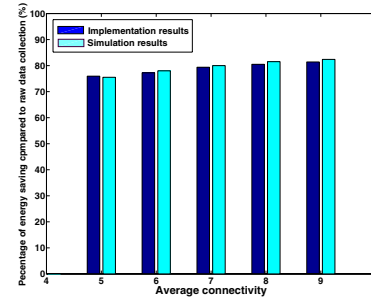


Figure 5. Percentage of energy saving compared to raw data collection in different average connectivity

The simulation results in Figure 5 show the energy-savings obtained when using DOSA as opposed to collecting raw data. The implementation results and simulation results are virtually identical. The in-depth analysis of the performance of a real-life implementation of DOSA can be found in [7].

6. Conclusion

This paper describe our experiences implementing and evaluating a distributed and self-organizing scheduling algorithm for energy-efficient data gathering on a 25-node, multihop sensor network. We show that while the performance of the algorithm in real-life is nearly identical to the theoretical results derived from simulations, a number of crucial steps that include creating a stable multihop network, managing the removal and addition of nodes, alleviating memory overflows and solving time synchronization, need to be taken to ensure the assumptions made in theory are applicable in real-life. Further work will focus on using statistical techniques to ensure more stable link qualities.

References

- [1] AIMS. 2006. Reef at our fingertips. <http://www.aims.gov.au/pages/about/communications/waypoint/headlines-04.html>.
- [2] Matlab. 2006. The mathworks-matlab-the language of technical computing. <http://www.mathworks.com/products/matlab/>.
- [3] N. Lynch, Distributed Algorithms. California: Morgan Kaufmann Publishers, 1996.
- [4] Ambient, Ambient systems, <http://www.ambientsystems.net/ambient/index.htm>, 2006.
- [5] L. Hoesel and P. Havinga, A lightweight medium access protocol (lmac) for wireless sensor networks, in *INSS*, Tokyo, Japan, June 2004.
- [6] Chatterjea, S. and Nieberg, T. and Meratnia, N. and Havinga, P.J.M, A distributed and self-organizing scheduling algorithm for energy-efficient data aggregation in wireless sensor networks, in *Technical Report TR-CTIT-07-10 Centre for Telematics and Information Technology, University of Twente*, 2007.
- [7] S. Chatterjea, Y.Zhang, T. Nieberg, and P. Havinga, Energy-efficient data acquisition using a distributed and self-organizing scheduling algorithm for wireless sensor networks, To be published *DCOSS*, 2007.
- [8] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, Directed diffusion for wireless sensor networking, *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 216, February 2003. [Online]. Available: <http://portal.acm.org/citation.cfm?id=638335>
- [9] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, Tinydb: an acquisitional query processing system for sensor networks, *ACM Trans. Database Syst.*, vol. 30, no. 1, pp. 122 173, 2005.
- [10] A. Deshpande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and W. Hong, Model-driven data acquisition in sensor networks, 2004. [Online]. Available: <http://citeseer.ist.psu.edu/706938.html>
- [11] D. Tulone and S. Madden, Paq: Time series forecasting for approximate query answering in sensor networks, in *EWSN*, 2006, pp. 2137.