Comparing wireless sensor network routing protocols

by Gerrit Niezen, Gerhard P. Hancke, University of Pretoria, and Imre J. Rudas, and László Horváth, Budapest Polytechnic, Hungary

Selection of a routing protocol for a wireless sensor network depends on various factors like the network lifetime, success rate and the number of nodes in the network. This article compares four popular routing protocols currently used in wireless sensor networks. The experimental setup is described, after which the various protocols are compared and evaluated.

A routing protocol is required when a source node can not send its packets directly to its destination node but has to rely on the assistance of intermediate nodes to forward these packets on its behalf. Although many protocols exist for traditional ad hoc wireless networks, none of them are suitable to the unique requirements of wireless sensor networks.

Routing protocols can be divided into two groups, proactive and reactive routing protocols [1]. With proactive routing a routing table is generated at each node, so that routing information is kept for every node in the network. This is used in traditional link state and distance vector routing protocols, with the destination sequenced distance vector (DSDV) protocol being proposed for wireless sensor networks.

With reactive routing no routing tables are generated and route discovery is done as needed. The route information is then kept for future reference. Existing reactive protocols include ad hoc on demand distance vector (AODV) and dynamic source routing (DSR). The problem with these protocols is that they minimise the total energy consumed in the network, but drain energy along paths chosen for minimum energy consumption. Network partitioning happens when some nodes in the network are drained from their battery energy more quickly than others, leading to some parts of the network being cut off from the rest of the network.

In this article the following routing protocols are simulated and evaluated:
- Flooding protocol - no routing tables are kept and messages are broadcast across the network.
- TinyGS Lx Multihop routing - a shortest path first algorithm with a single destination node (the sink node) and active two way link estimation.
- Low energy adaptive clustering hierarchy (LEACH) - a clustering-based protocol [2], and
- Ad hoc on demand distance vector (AODV) - a reactive routing protocol [3].

Experimental setup

Simulation settings

The various routing protocols were compared through the simulation of wireless sensor networks with the OMNeT++ discrete event simulation system [4]. In [5] it is stated that traffic generation should be based on intended applications. Since wireless sensor networks are usually designed to handle sporadic or slowly changing events (such as temperature and pressure variations) [6], it was assumed that the application layer transmits information every 15 minutes. In table 1 the settings used during the simulation are displayed. For the radio model, a transmission distance of 1 m was used, while the transmitter power was set at 2 mW on the mobility framework’s physical layer implementation.

Calculating power consumption

To calculate power consumption, the simple radio model introduced by [7] was used. It was initially used by the authors to calculate the power consumption of the LEACH protocol and was implemented in the OMNeT++ simulator by [81].

Energy consumed during transmission ($E_{tx}$) is calculated with

$$E_{tx} = E_{elec} + E_{ampl}kd^2$$

where $E_{elec}$ is the energy consumed by the radio transceiver, $k$ is the size of the message in bits, $E_{ampl}$ is the energy consumed by the radio amplifier and $d$ is the transmission distance in metres.

It should be noted that even if the nodes are mobile, the transmission distance is still constant, as the transmission power of the radio transceiver does not change (except in the case of LEACH where the transmission power is varied to reach the base station node).

The energy consumed during message reception is calculated by

$$E_{rx} = E_{elec}k$$

where $E_{elec}$ is the energy consumed by the radio transceiver and $k$ is the size of the message in bits. Energy consumption during reception is only calculated when a message is received, i.e. the radio transceiver only expends power during message reception. This is usually not the case, as a radio transceiver still expends a large amount of power when in receive mode and not receiving messages. It is therefore assumed that a time synchronisation protocol is implemented to switch off the radio transceiver when messages are not being sent or received.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmission distance</td>
<td>$d=1$ m (for power calculations)</td>
</tr>
<tr>
<td>P$_{max}$</td>
<td>2 mW (transmitter power)</td>
</tr>
<tr>
<td>Bit rate</td>
<td>10 kbps</td>
</tr>
<tr>
<td>PNRG used and seed</td>
<td>Mersenne Twister</td>
</tr>
<tr>
<td></td>
<td>Seed automatically generated from OMNeT++ simulator</td>
</tr>
<tr>
<td>Simulation package and version number used</td>
<td>OMNeT++ Discrete Event 3.2p1 with Mobility Framework 2.0p2</td>
</tr>
<tr>
<td>Traffic type</td>
<td>Data message sent once every 15 minutes</td>
</tr>
<tr>
<td>Mobility model</td>
<td>Static, with random node placement</td>
</tr>
</tbody>
</table>

Table 1: Simulation settings.
The assumption is made that the radio channel is symmetric, i.e. the energy consumed to transmit a message from node A to node B is the same as to transmit a message from node B to node A.

Experimental setup of flooding protocol

The flooding protocol can be seen as the most basic type of routing protocol, where no routing tables are kept and messages are broadcast across the network. It is also the least energy efficient way to send messages across a wireless network. The signal attenuation threshold for the simulation was set at 110 dBm and a carrier frequency of 868 MHz was used.

Experimental setup of TinyOS 1.x multihop routing

TinyOS 1.x multihop routing is a shortest-path-first algorithm with a single destination node (the sink node) and active two way link estimation. The TinyOS Surge application is an implementation of this algorithm, with an application layer that transmits sensor information from a simulated analogue to digital converter (ADC) at predetermined intervals.

A language translator called NesCT [9] was used to convert the TinyOS source code (written in the TinyOS language NesC) to OMNeT++ source code. NesCT has the capability to allow the TinyOS source to be translated into the format required by the mobility framework.

The mobility framework is a framework to support wireless and mobile simulations in OMNeT++. It includes basic components for the application layer, network layer, NIC (network interface controller) or physical layer, media access control (MAC) layer and also includes components for simulating mobility. For this study static nodes were used and mobility was not implemented.

A simulation’s playground size can be defined as the simulated area in which the network is contained. A playground size of 200 by 200 units specifies a rectangular area in which nodes can have X and Y coordinates from zero up to 200 units.

Static nodes with random node placement were used. The Mersenne Twister pseudo random number generator was used to place the nodes, with the seed generated automatically from the OMNeT++ simulator.

To ensure that the nodes will be close enough to one another to be within transmission range, the playground sizes were simulated as specified in table 2. If the nodes are too close to one another, most of the time will be spent contending for the available wireless spectrum, and the nodes will find it difficult to actually transmit data.

For the simulations a distinction was made between routing update messages and data messages. Routing update messages were defined to have a size of 64 bits, while data messages were defined to have a size of 105 bits. Nodes were initialised with only 5 mJ of energy to speed up the simulation.

Experimental setup of LEACH

LEACH is a self-organising, adaptive clustering protocol that uses randomisation to distribute the energy load evenly among the sensors in the network. In LEACH, the nodes organise themselves into local clusters, with one node acting as the local base station or cluster head. If the cluster heads were fixed throughout the system lifetime, as in conventional clustering algorithms, the sensors chosen to be cluster heads would die quickly, shortening the lifetime of the network dramatically. Thus LEACH includes randomised rotation of the cluster heads such that it rotates among the various sensors in order to not drain the battery of a single sensor [7]. As seen in Fig. 1, there is an initial set-up phase for each round during which the cluster heads are chosen, and a steady-state state during which each node is allocated a slot for data communication.

LEACH uses a distributed algorithm to determine the cluster heads during the set-up phase. When clusters are created, each node autonomously decides if it will be a cluster head for the next round. The selection is made stochastically, in that each node selects a random number between 0 and 1 and if the number is lower than a threshold $T(n)$, the node becomes a cluster head. $T(n)$ is determined by nodes that have not been cluster head in the last $1/P$ rounds, otherwise $T(n)$ is zero [10].

$P$ is a predefined number that determines the average number of cluster heads during a round and $r$ is the number of the current round.

$T(n) = \frac{P}{1 - P \left( r \mod \left( \frac{1}{p} \right) \right)}$

SolarLEACH [LO] is a solar aware implementation of the LEACH protocol. For this study the SolarLEACH implementation was used with the solar option disabled, in order to test the default distributed LEACH protocol. A centralised version is also available in the implementation. The SolarLEACH implementation is optimised for speed on the OMNeT framework, but is very experimental and prone to crashing. Due to its experimental nature it was only possible to simulate network sizes up to 100 nodes.

Experimental setup of AODV

The TinyAODV stack implements a simplified version of the ad hoc on demand vector (AODV) routing algorithm that is implemented in TinyOS. As with the LEACH implementation, NesCT was used to convert the TinyOS code into OMNeT code. It is a reactive algorithm, so it builds routes on demand when desired by source nodes. It builds routes using route request/route-reply messages. A source node desiring a route to the destination generates and broadcasts a route request (RREQ) message across the network. When the RREQ arrives at the destination or an intermediate node with the path to the destination, a route replay (RREP) message is generated and propagated along the reverse path. The nodes propagating the RREP back to the source add a route entry for the destination. It supports many RFC features, but it takes into account the reduced capabilities of the wireless sensor network platforms operating on TinyOS.

The implementation of TinyAODV makes various simplifications in order to implement the AODV protocol in a very small footprint:

- RREP messages are only generated by the destination. Routes never expire.
- Only the hop count metric is used.
- No messages are generated to keep routes active because routes never expire. Route errors are generated when a data message can no longer be sent over the path. This is detected using the TinyOS link level acknowledgments.

Unfortunately the TinyAODV implementation by NesCT only allows a network size of four nodes. This makes a test like determining the number of active nodes in the network at certain intervals redundant. For this report the average number of messages sent in the network was simulated and evaluated.
Results

Time until the first node fails

The time until the first node fails (with the results for TinyOS Lx multihop routing) can be seen in Fig. 5. For a small network of only 10 nodes, the first node dies just before $t = 60$ minutes. This is a huge improvement compared to the flooding protocol, where the first node in the network dies after only 15 minutes, just after the first data packet has been sent (see Fig. 2). It does get worse for larger network sizes, where the first node dies just after 20 minutes. This is due to the larger number of packets that have to be forwarded in the network.

In the case of LEACH (seen in Fig. 3) the time until the first node fails is critical, as it is a cluster-based protocol and the node that fails will most probably be a cluster head. Unfortunately, for a network size of 100 nodes, the first node fails after 15 minutes, even before the first data packet has been sent. This is due to the large amount of messages required during the initial setup period, during which the cluster heads are chosen.

Both TinyOS Lx multihop routing and LEACH perform about three times better than the flooding protocol. While the performance of these two routing protocols is similar with regards to the first node failing, the TinyOS algorithm performs better for larger network sizes. The cluster heads in LEACH fail earlier since they have to forward more data than the regular nodes in the network and due to the extra initial setup packets required during each round.

Average remaining energy in the network at selected intervals

In Fig. 5 the average remaining energy of all the nodes in the network are displayed at selected time intervals for the flooding protocol. At $t = 15$ minutes, the first data packet is sent and the power for each node is still close to maximum. After 16 minutes, most of the nodes have no remaining energy and just a few nodes have enough power for the second data packet transmission.

In Fig. 6 the average remaining energy in the network at selected intervals for TinyOS Lx multihop routing can be seen. Network sizes of both 10 and 50 nodes were simulated to be able to compare the average remaining energy for different network sizes. In both cases almost all the nodes have died after about 60 minutes. Comparing this to Fig. 4 it can be seen that the first node died just before $t = 60$ minutes. This indicates that there is an almost even energy consumption distribution across the network, where most of the nodes die during the same time interval (from $t = 60$ to $t = 75$ minutes).

The average remaining energy for 10 nodes is linear for the selected intervals, while the remaining energy for 50 nodes can almost be fitted to an exponential curve.

To determine the average remaining energy in the network at selected intervals for LEACH as shown in Fig. 7, a network size of 50 nodes was used. There is a small linear decrease as time progresses, which can be accounted for when considering the power requirements of every setup period during each of the intervals.

While the flooding protocol has almost no remaining energy after the first round of data messages at $t = 15$ min, the TinyOS algorithm tends to drain its energy after about 60 minutes.

In this instance the LEACH protocol performs much better, with the average remaining energy at $t = 75$ min still at around 75%.

Time until the sink is unreachable, due to all of its neighbours failing

In the case of the flooding protocol, the sink node becomes unreachable after just 15 minutes. For the TinyOS Lx multihop routing protocol, the different network sizes do not seem to have an effect on the time until the sink is unreachable, which is around 75 minutes. The results were the same for network sizes of 50, 100 and 250 nodes.

For the LEACH protocol, such a test does not make sense, since this will be the instant when all the nodes have died. Because all the nodes are within range of the base station (depending on their adjustable transmission range), the cluster heads will be seen as sink neighbours.
With rotating cluster heads, all the nodes in the network should be regarded as sink neighbours.

Number of active nodes in the network at certain intervals

As seen in Figs. 8 and 9, the LEACH protocol tends to perform better than the TinyOS protocol with regards to the number of active nodes in the network. LEACH has 90% active nodes at t = 125 minutes, while the TinyOS algorithm only manages t = 38 minutes. For 25% active nodes, LEACH performs almost an order of magnitude better than the TinyOS algorithm. It is interesting to note that TinyOS multihop routing performs better for smaller size networks, whereas LEACH actually performs better for larger networks. For a network of 100 nodes, LEACH still has 25% active nodes left after t = 417 minutes, compared to t = 291 minutes for the network size of 50 nodes.

With the flooding protocol, almost no nodes remain active after the first data packet is sent. All the other nodes' energy is expended just by forwarding data messages from other nodes in the network.

Average number of messages sent in the network at selected intervals

In Fig. 11 the average number of messages sent at selected intervals with LEACH are shown, for a network size of 50 nodes. The linear increase indicates that while there is a greater chance of the cluster heads dying earlier, the rest of the nodes in the network continue to transmit data. This is due to most of the nodes not being required to forward data in the network, as this is handled by the cluster heads.

The average number of messages sent (i.e. number of messages sent per node) is very high for the AODV routing protocol compared to the other routing protocols, as seen in Fig. 10. Using the 5mJ initial energy applied for the other simulation models, the simulation only ran for a minute before all nodes were drained of their energy. The network size for the AODV simulation was set at 4 nodes.

A single round of data messages sent with the flooding protocol requires about 160 messages sent per node on average. While the TinyOS algorithm sends more messages initially (50 messages vs. 28 messages) than the LEACH protocol, the LEACH protocol sends more messages in total due to the cluster head setup periods during each round.

The difference however, is marginal: 150 messages sent by the TinyOS algorithm vs. 160 messages sent by the LEACH protocol. With the AODV protocol, almost 3500 messages have been sent on average after only 15 minutes. LEACH has 90% active nodes at t = 125 minutes, while the TinyOS sink has only received 27 messages per node, while the LEACH sink has already received 110 messages per node.

For the flooding protocol, more than 500 messages were received by the sink node after the first packet data transmission. This means that for every node in the network that sent a message, more than nine duplicates were received by the sink node. Since most of the nodes between the sink and the outer edges of the network died after the first data packet transmission, none of the data messages sent during the second interval were received by the sink node.

Conclusion

While allocating only 5mJ of energy to each node at startup is a bit unrealistic, it shows the tremendous amount of energy expended for even one data transmission interval in the network when using a flooding protocol.

While the forwarding rules are simple, their performance in terms of sent packets, or delay, is very poor. These shortcomings are due to ignoring the network's topology. Simulation results for the flooding protocol were validated against the original OMNeT++ simulations done by [8].

The playground size (i.e. the size of the area in which the nodes are placed) has a large effect on the outcome of the simulation. If the nodes are distributed randomly, the playground size has to be small enough so that enough nodes can communicate with one another. It should not be so small so that the nodes are constantly in contention for the available radio spectrum, in which case they will not be able to communicate if the radio channels are not available.

It should be noted that the credibility of the presented results depend on the quality of the software used to implement the protocols.

Experiments with the TinyOS Lx multihop routing protocol suggest that the network takes a long time to converge, even for a small number of nodes. The topology only starts to build after one or two minutes. By default, the module sends a route update message once every 10 seconds and re computes after 50 seconds (i.e. 5 route
LEACH makes some assumptions about both the sender nodes and the underlying network, e.g., assuming that all sensor nodes can adapt their transmission range and that energy consumption during transmission scales exactly with distance [10].

It is also assumed that every sensor node will be able to reach the base station if it increases its transmission range. In a typical wireless sensor network this is not the case.

Since AODV was originally developed for wireless networks such as WiFi, it does not conform to the power requirements of wireless sensor networks. While it can be implemented in a wireless sensor network setting (TinyAODV runs on any node that employs TinyOS), the routing protocol will require serious adjustments and fine-tuning to minimise its power requirements. The average number of messages sent in the network was much higher than any of the other protocols, including the flooding protocol.

Selecting a routing protocol to use in a wireless sensor network is dependent on various factors like the network lifetime, success rate and the number of nodes in the network. The protocols evaluated in this article performed better in some of the criteria than others, but none of the protocols were able to fulfil all of the stated criteria. The TinyOS Lx multihop routing protocol performed better than the LEACH protocol for larger network sizes, whereas the LEACH protocol had a better success ratio for the number of messages received by the sink node. LEACH had a longer network lifetime and average remaining energy, but the first node failed earlier than in a TinyOS Lx multihop routing network.

To select a routing protocol for a wireless sensor network, attention must be paid to the type of environment the network is going to be deployed in, the quality of service required with regards to the sensed data and the size of the network. The capabilities of the physical sensor nodes, such as if the transmission distance can be scaled, should also be considered.

Trying to adapt traditional wireless MANET (mobile ad-hoc network) protocols that were not originally designed with the energy requirements of wireless sensor networks in mind, does not seem feasible after evaluating a routing protocol like AODV.

References


Acknowledgement

This paper was presented at the IEEE Africon 2007 conference in Windhoek, Namibia and is republished with permission.

Contact Prof. Gerhard Hancke, University of Pretoria, Tel 012 420-2386, gerhard.hancke@eng.up.ac.za