Simulation-Based Performance Analysis Of Wireless Routing Protocols For Smart Metering Applications

Peter Sooki
Akos Korosi
Consortium of DataShop Ltd. and GATE Non-profit Ltd. - Hungary

doi: 10.19044/esj.2016.v12n9p1 URL:http://dx.doi.org/10.19044/esj.2016.v12n9p1

Abstract

There is a growing interest for smart metering applications in wireless environment recently. Wireless smart meter devices are easy to use solutions for recording energy consumption and report back the amount to the utility on a daily basis. This kind of usage of smart meter devices can be considered as a special subset of wireless sensor networks (WSN). The communication protocol used in smart meter networks has to be fast, reliable and secure. The challenge in developing such a protocol is the difficulty to debug software on multiple wireless nodes at the same time. This difficulty highlights the importance of the simulation-based approach. The most laborious part of the wireless communication is the routing protocol in the network layer when it comes to implementation. In this paper different wireless communication routing protocols are compared by simulation including naive Flooding, Gradient-Based Routing (GBR) and Directed Diffusion (DD). The simulation results show that GBR and DD have a superior performance over flooding as expected, and that GBR and DD has similar efficiency in a smart metering application. Since GBR is easier to implement it is the recommended solution for routing in a smart metering network.

Keywords: Smart metering, WSN, routing protocols, flooding, gradient-based routing, directed diffusion

Introduction

Smart meters are electronic energy consumption measurement devices gathering energy usage information in domestic environment.

A smart meter can send back the collected information to the public utility for billing and monitoring without manual intervention facilitating the service for the consumer and the utility as well. Using this feature the utility is able to build a more sophisticated picture about the consumption habits, and the customer can also be instantly informed about the consequences of different energy usage. As another benefit, smart meters have two-way communication channel to the central system, and therefore are capable of displaying energy costs with current prices to the customer. The evolution of microelectromechanical systems (MEMS) allowed the manufacturing of low consumption, low cost, small sized sensor nodes responsible for measurement, data collection, processing and data transferring. The interoperability of such nodes form a wireless sensor network (WSN) [5]. In a wireless sensor network formed by smart meter devices the nodes are communicating with a remote server through a data concentrator device. The data concentrator connects the individual nodes data channels to a single destination over Internet. The nodes are usually organized into a centralized network such as mesh network, where there are self healing and redundant data paths. The most vital requirements against the centralized network are:

- precise time synchronization capability between the nodes
- data encryption
- tamper resistance
- cost effective data routing

Regarding the first three point one can use industry standard solutions most of them available in low-level implementation in microcontroller or SOC level environment. The challenge comes in when the need for implementing and debugging a specific routing protocol arises, knowing that the wired routing protocols are less efficient in a multi-hop wireless environment. Nowadays there are well tested standard routing protocols such as AODV (Ad Hoc On-Demand Distance Vector routing) used in ZigBee or DSDV (Destination-sequenced Distance-Vector routing), however these are general purpose and heavyweight WSN routing protocols not considering the special needs and features of smart meter devices and networks. A smart metering network differs from an average wireless sensor network in many ways:

- the nodes have a fixed location
- usually there are fewer than 100 nodes in a network
- the nodes are not battery operated
- there is a data concentrator responsible for data aggregation

The emerging need for simple and robust routing protocols for smart meter networks are driven by these differences. The routing protocols in wireless sensor networks can be classified into flat, hierarchical and location based protocols (Al-Karaki, 2004; Arampatzis, Th.; Lygeros, J.; Manesis, 2005).

In flat routing all the nodes are similar and have the same task assigned. The routing is performed in a multi-hop manner in case of flat routing. Routing protocols in the flat architecture include: Flooding, Directed Diffusion (DD), Gradient-based Routing (GBR), Cougar, Spin, Minimum Cost Forwarding Algorithm.

In hierarchical routing the nodes are differentiated by their roles. The network is organized into clusters, where each node responds only to its respective cluster-head, which is responsible for data aggregation, data transmission to the outside world and cluster management. The hierarchical routing approach lowers the traffic and latency on a network. Routing protocols in hierarchical routing include: Low-Energy Adaptive Clustering Hierarchy (LEACH), Power-Efficient Gathering in Sensor Information Systems (PEGASIS).

The location based routing takes advantage of knowing the position of the nodes and having signal strength information from the packet transferring actions. Routing protocols in location based routing include Geographic Adaptive Fidelity (GAF).

This paper will analyse the performance of flat routing protocols in a smart metering device network on a simulation-based approach including Flooding, Directed Diffusion and Gradient-based Routing.

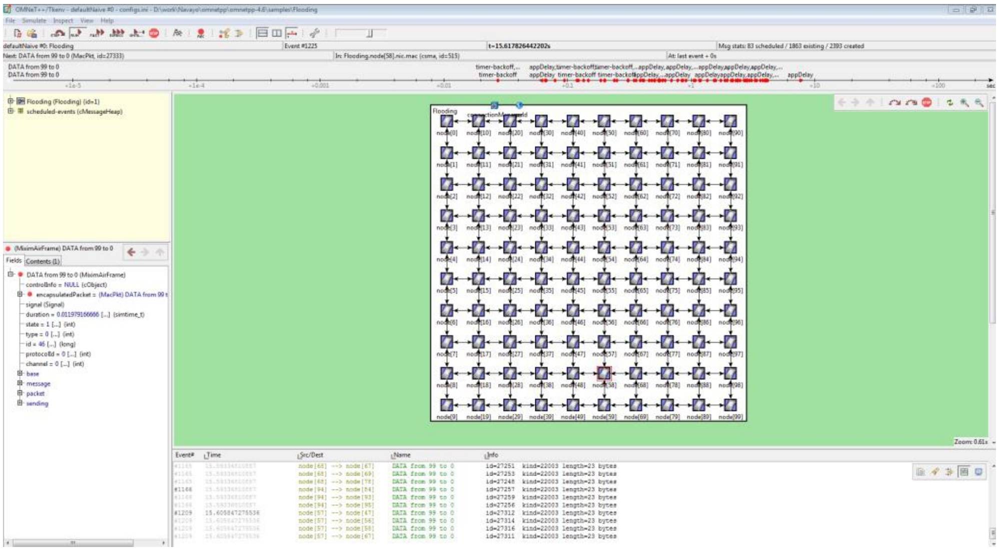
This paper organized as follows. Section 2 presents the simulation framework and methods, Section 3 describes the tested protocols, Section 4 presents the simulation results and Section 5 concludes the paper.

Simulation framework and methods

Several simulation experiments were conducted with the OMNet++ simulator during the work. OMNeT++ is a modular, component-based C++ simulation library and framework suitable for building network simulations. An OMNet++ simulated network is built upon hierarchical modules, where the depth of the modules are optional. The modules inside the framework are communicating with messages, that can be complex data structures. The messages can be sent directly to a module or through different channels. The user is responsible for defining the behaviour of the modules at the lowest layer in the hierarchy using C++ language. The framework supports the module development with built-in software libraries. The user defined modules are running in parallel as coroutines during the simulation.

The framework comes with different user interfaces to facilitate debugging, running batches and demonstration.

Figure 1. Simulation view of OMNet++


OMNeT++/Tkenv - defaultNaive #0 - configs.ini - D:\work\Navayo\omnetpp\omnetpp-4.6\samples\Flooding
File Simulate Inspect View Help
Flooding
connectionManager
Zoom: 0.61x

```
defaultNaive #0: Flooding        Event #1225        t=15.617826442202s        Msg stats: 83 scheduled / 1863 existing / 2393 created
Next: DATA from 99 to 0 (MacPkt, id=27333)        In: Flooding.node[58].nic.mac (csma, id=515)        At: last event + 0s
```

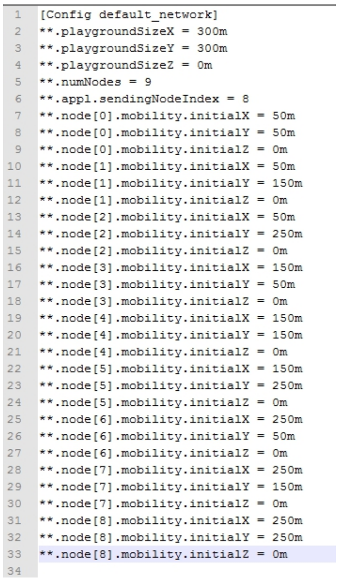
```
Flooding (Flooding) (id=1)
scheduled-events (cMessageHeap)
```

```
(MiximAirFrame) DATA from 99 to 0
Fields  Contents (1)
DATA from 99 to 0 (MiximAirFrame)
  controlInfo = NULL (cObject)
  encapsulatedPacket = (MacPkt) DATA from 99 t
  signal (Signal)
  duration = 0.011979166666 [...] (simtime_t)
  state = 1 [...] (int)
  type = 0 [...] (int)
  id = 46 [...] (long)
  protocolId = 0 [...] (int)
  channel = 0 [...] (int)
base
message
packet
sending
```

```
Event#  Time              Src/Dest                 Name                Info
#1165   15.583368100887   node[68] --> node[67]    DATA from 99 to 0   id=27251  kind=22003 length=23 bytes
#1165   15.583368100887   node[68] --> node[69]    DATA from 99 to 0   id=27253  kind=22003 length=23 bytes
#1165   15.583368100887   node[68] --> node[78]    DATA from 99 to 0   id=27248  kind=22003 length=23 bytes
#1168   15.583368100887   node[94] --> node[84]    DATA from 99 to 0   id=27257  kind=22003 length=23 bytes
#1168   15.583368100887   node[94] --> node[93]    DATA from 99 to 0   id=27259  kind=22003 length=23 bytes
#1168   15.583368100887   node[94] --> node[95]    DATA from 99 to 0   id=27256  kind=22003 length=23 bytes
#1209   15.605847275536   node[57] --> node[47]    DATA from 99 to 0   id=27312  kind=22003 length=23 bytes
#1209   15.605847275536   node[57] --> node[56]    DATA from 99 to 0   id=27314  kind=22003 length=23 bytes
#1209   15.605847275536   node[57] --> node[58]    DATA from 99 to 0   id=27316  kind=22003 length=23 bytes
#1209   15.605847275536   node[57] --> node[67]    DATA from 99 to 0   id=27311  kind=22003 length=23 bytes
```

The interconnection of different modules is defined using a special high level language NED (NEtwork Description). The framework translates the NED modules to C++, and there is also an opportunity to graphically define the network with the GNED extension. The simulation parameters can be defined in .ini files before running the actual simulation process.

Fig. 2. shows an .ini file generated with a Python script. The .ini file defines 9 coordinates in a grid with 100m distance between the nodes.



```
[Config default_network]
**.playgroundSizeX = 300m
**.playgroundSizeY = 300m
**.playgroundSizeZ = 0m
**.numNodes = 9
**.appl.sendingNodeIndex = 8
**.node[0].mobility.initialX = 50m
**.node[0].mobility.initialY = 50m
**.node[0].mobility.initialZ = 0m
**.node[1].mobility.initialX = 50m
**.node[1].mobility.initialY = 150m
**.node[1].mobility.initialZ = 0m
**.node[2].mobility.initialX = 50m
**.node[2].mobility.initialY = 250m
**.node[2].mobility.initialZ = 0m
**.node[3].mobility.initialX = 150m
**.node[3].mobility.initialY = 50m
**.node[3].mobility.initialZ = 0m
**.node[4].mobility.initialX = 150m
**.node[4].mobility.initialY = 150m
**.node[4].mobility.initialZ = 0m
**.node[5].mobility.initialX = 150m
**.node[5].mobility.initialY = 250m
**.node[5].mobility.initialZ = 0m
**.node[6].mobility.initialX = 250m
**.node[6].mobility.initialY = 50m
**.node[6].mobility.initialZ = 0m
**.node[7].mobility.initialX = 250m
**.node[7].mobility.initialY = 150m
**.node[7].mobility.initialZ = 0m
**.node[8].mobility.initialX = 250m
**.node[8].mobility.initialY = 250m
**.node[8].mobility.initialZ = 0m

```

Figure 2. Ini file defining node coordinates

On the top of the OMNet++ simulation engine the MiXiM (mixed simulator) framework were used for the wireless network routing simulation. The MiXiM project incorporates several OMNet++ framework written to support mobile and wireless simulations.

The predecessors are:

- ChSim (Universitaet Paderborn)
- Mac Simulator (Technische Universiteit Delft)
- Mobility Framework (Technische Universitaet Berlin, Telecomm Networks Group)
- Positif Framework (Technische Universiteit Delft)

MiXiM offers detailed models of radio wave propagation, interference estimation, radio transceiver power consumption and wireless MAC protocols.

Application examples of MiXiM framework:

- Wireless sensor networks
- Body area networks
- Ad-hoc networks
- Vehicular networks

Description of the tested protocols

There are many protocols proposed to solve the routing in wireless sensor networks. As described in the introduction they follow different approaches and architecture. This section explores the implementation of the concerned protocols.

Naive flooding

The family of flooding algorithms is the simplest approach to solve the routing in a flat architecture WSN. In uncontrolled naive flooding there is no addressing, instead all the received messages are continuously broadcasted by the nodes. In naive flooding there are no directions, the nodes receives the same packages from their neighbours repeatedly. The naive solution only works correctly if the nodes are organized into a spanning tree. In case the topology of the network contains a circle, the packets will travel around this loop indefinitely. If there are two or more loops in the topology, then the messages are duplicated while circle around, until they eat up all the available bandwidth (broadcast storm).

The controlled flooding protocols addresses the aforementioned issues. In Sequence Number Controlled Flooding (SNCF) every node attaches its own address and sequence number to the message, and stores the packet in its own dedicated memory. If the node receives the same packet again, it can be dropped immediately.

In Reverse Path Forwarding (RPF) the nodes only broadcast messages in the forward direction. If the packet is received form the next node, the the packet is returned to the original sender. The nodes are only broadcasting a packet when the message is arriving on the same route as the known return path. To use RPF each node should have a path to every other nodes.

Flooding has the following strengths:

- does not rely on setup or topology maintenance
- resistant to malfunctioning nodes
- will route messages to the destination if a path to the destination exists
- simple implementation
- the shortest path will be used (among the others)

On the other hand flooding has the following weaknesses:

- not energy efficient
- huge communication overhead

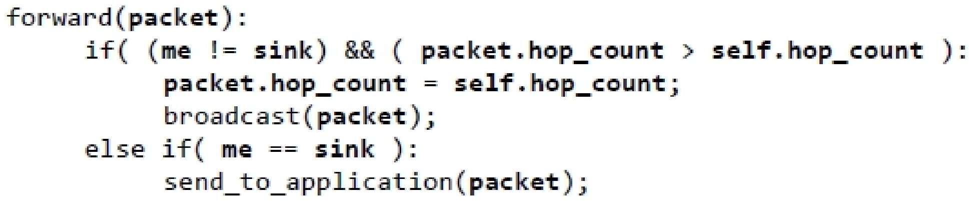
Gradient-Based Routing (GBR)

Flooding cannot be considered to be an efficient solution when the aim is to deliver packets to a single sink, or data concentrator in the case of a smart metering network. GBR offers a solution, where a packet is only transmitted by a receiver node when the distance of the sender from the data concentrator is higher than the receiver's distance. The algorithm is based on the assumption that each node knows its own distance from the concentrator expressed in hop count, where the hop count refers to the number of intermediate nodes through which packets must pass to reach the destination. In order to discover the node distances / gradients the data concentrator must initiate an identification phase.

The identification phase has the following steps (Yoo-Shim- Kim, 2011):

1. Initially, each sensor node sets its gradient to infinity. $R \to \infty$.
2. The drain initiates a network build up phase by broadcasting an advertisement packet containing its own cost, which is zero (Rsink = 0).
3. When a node receives an advertisement packet it compares the received gradient and the cost of the link to its current gradient.
4. If the newly acquired gradient and the link cost is smaller than the old one, the node stores the new value and broadcasts an advertisement packet with the new gradient.

The cost of the link can be calculated considering the signal strength and the energy level of the sender node. A data packet in GBR consists of the gradient of the sender, the cost of the link and the payload. After receiving a packet each node performs the following simple algorithm:



```
forward(packet):
      if( (me != sink) && ( packet.hop_count > self.hop_count ):
            packet.hop_count = self.hop_count;
            broadcast(packet);
      else if( me == sink ):
            send_to_application(packet);
```

The concentrator cannot be notified about a fallen out node in an easy way. A periodically initiated identification phase can solve this problem. In an event triggered solution the concentrator is able to notice missing data from the malfunctioning node, and thus initiate a network build up phase.

Directed Diffusion (DD)

The Directed Diffusion (DD) is a data centric protocol, where data is described by attribute-value pairs. In a typical DD application there is only one sink, a data concentrator, which aggregates data from the nodes. In a path discovery phase the sink disseminates an interest broadcast message throughout the sensor network. This process sets up gradients within the network, where a gradient can be the sum of the attribute-value pair and the direction. The magnitude of the gradient can be different, depending on the neighbouring nodes. The data from the nodes starts flowing in the direction of steepest ascent (Al-Karaki - Kamal, 2004).

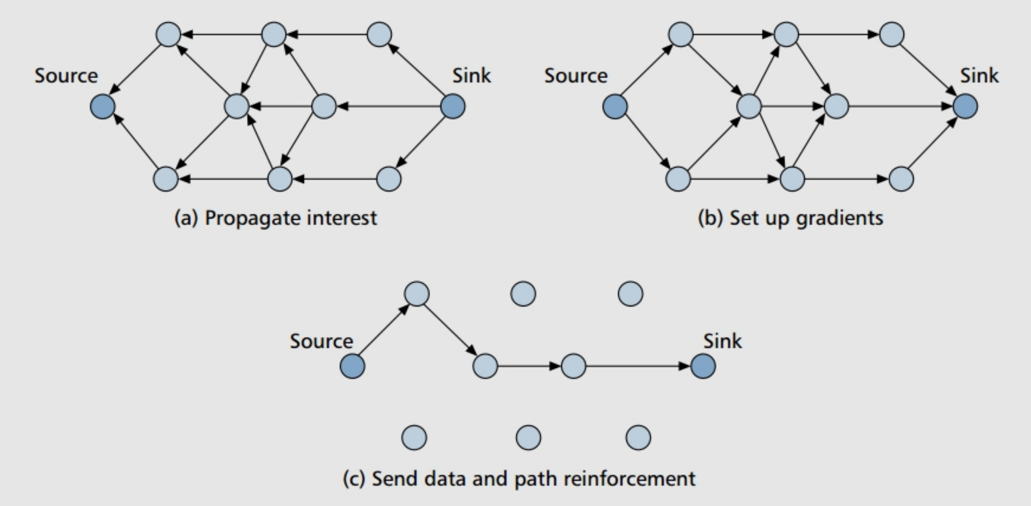


Figure 3. Packet propagation in a network using Directed Diffusion

The interest broadcast message is repeated periodically by the sink. The aim is to build a spanning tree structure. The Directed Diffusion has a disadvantage that nodes in advantageous positions are used more frequently. To overcome this drawback load balancing algorithms are used.

Simulation results and discussion

4.1 The simulations were conducted using grid and row topology with different node counts. The distance between nodes were determined in a way that only immediate neighboring nodes were within radio range.

In each case the two most remote nodes, a sender and a receiver were communicating with periodically sent messages. All the other nodes were only transmitting the received messages.

The following performance metrics were defined.

- #of nodes: number of nodes
- min. hop count: the minimum number of intermediate nodes through which packets passed to reach the destination
- @10msgs avg hop count: the average hop count for the first 10 messages
- @10msgs total TX: how many messages were transferred in the network during the arrival of the first 10 messages. This metric shows the load on the network caused by the algorithm.
- @1 arrival time: the arrival time of the first message (sec).

ROW topology

The row topology is not a realistic approach, the only benefit here is getting familiar with the basic features.

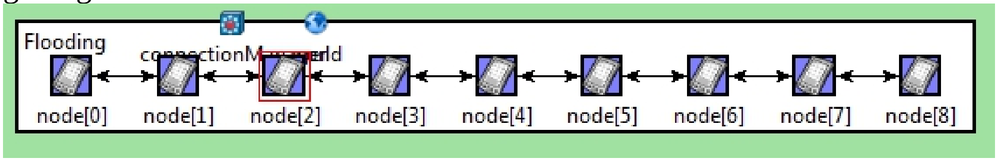


Figure 4. Row topology

In the following tables the simulation results of the examined routing algorithms are collected in case of a row topology.

#of nodes	min hop count	@10msgs avg hop count	@10msgs total TX	@1 arrival time
2	1	1	10	0.3124791667
9	8	10.6	213	3.4198333333
49	48	50.6	2994	19.199
100	99	101.8	10861	42.9954374999

Table 1. Naive Flooding results

#of nodes	min hop count	@10msgs avg hop count	@10msgs total TX	@1 arrival time
2	1	1	10	0.3124791667
9	8	8	80	3.1731666667
49	48	48	499	17.439
100	99	99	1184	40.5279375

Table 2. Gradient-based Routing results

#of nodes	min hop count	@10msgs avg hop count	@10msgs total TX	@1 arrival time
2	1	1	19	0.3124791667
9	8	8	152	3.1731666667
49	48	48	931	17.439
100	99	99	1746	40.5279375

Table 3. Directed Diffusion results

The following tables show the comparison of results.

#of nodes	min hop count
2	1
9	8
49	48
100	99

Table 4. Minimum hop counts

It was only possible for a packet to propagate through the network when it hit every single node because of the row arrangement. The “min. hop count” shows that in each case there was at least one package which travelled through the straight path.

avg hop_count @10 recv			
#of nodes	NAIVE	GRADIENT	DIFFUSION
2	1	1	1
9	10.6	8	8
49	50.6	48	48
100	101.8	99	99

Table 5. Average hop counts

Table 5. shows that the average hop count coincide with the minimum hop count in case of the little more sophisticated algorithms. Considering naive flooding the messages are transmitted backward as well, causing worse average hop count than the ideal. The underlying radio signal propagation model can also be considered here.

total netwk tx @10 recv			
#of nodes	NAIVE	GRADIENT	DIFFUSION
2	10	10	19
9	213	80	152
49	2994	499	931
100	10861	1184	1746

Table 6. Total number of messages

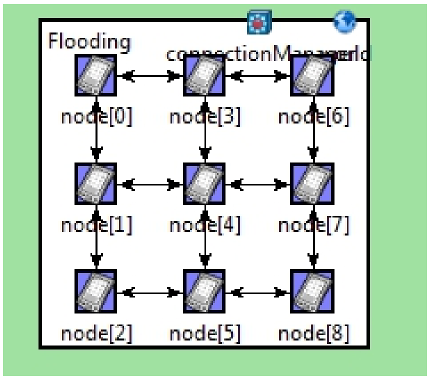
In the naive case the total number of transmitted messages in the network increases nearly exponentially, while the GBR and DD algorithms shows a reduced growing rate. It may seem that the diffusion is worse than GBR as it shows twice the amount of transmitted messages. To explain this symptom the routing table in each node should be considered in case of directed diffusion. As the nodes should maintain their routing tables, an acknowledgement message is delivered to each sender after the reception of a packet. This process implies twice as much messages in overall. Despite the fact that this is a drawback in a particular case, the two algorithms are within the same order of magnitude.

It can be noticed that the total number of messages in case of DD is not exactly twice as much as in the gradient case. This is because the simulation stop condition - the arrival of ten messages - has been reached before all the acknowledgement packets arrived back to the sender node, although this does not substantially influence the results.

GRID topology

The grid topology is a more realistic approach, as it well approximates an apartment building environment, where smart meter devices can be located similarly, equally spaced to each other. It is also realistic that the devices are within the range when they are immediate neighbours as shown in Fig. 5.

Figure 5. Grid topology



#of nodes	min hop count	avg hop count	@10 total netw TX	@1 arrival time
2	1	1	10	0.3324791667
9	4	4.8	130	1.3299166667
49	12	12.6	1257	3.34975
100	18	18.8	3241	5.344625

Table 7. Naive Flooding results

#of nodes	min hop count	avg hop count	@10 total netw TX	@1 arrival time
2	1	1	10	0.3324791667
9	4	5.4	70	1.3299166667
49	12	12	735	3.34975
100	18	18.6	2106	5.344625

Table 8. Gradient-based Routing results

#of nodes	min hop count	@10 avg hop count	@10 total netw TX	@1 arrival time
2	1	1	10	0.3324791667
9	4	5.6	79	1.3299166667
49	12	12	827	3.34975
100	18	18	2566	5.344625

Table 9. Directed Diffusion results

The following tables show the comparison of results.

#of nodes	min hop count
2	1
9	4
49	12
100	18

Table 10. Minimum hop counts

The shortest path through the network in the proposed grid arrangement is the sum of the width and height of the grid minus two. The “min. hop count” shows that in each examined case there was at least one package which travelled through the shortest path.

avg hop_count @10 recv			
#of nodes	NAIVE	GRADIENT	DIFFUSION
2	1	1	1
9	4.8	5.4	5.6
49	12.6	12	12
100	18.8	18.6	18

Table 11. Average hop counts

The average hop count well approximates the ideal situation in case of GBR and DD, while the naive approach is getting worse as the number of nodes increases.

total netwk tx @10 recv			
#of nodes	NAIVE	GRADIENT	DIFFUSION
2	10	10	19
9	130	70	79
49	1257	735	827
100	3241	2106	2566

Table 12. Total number of messages

The GBR and DD outperforms the naive algorithm in the total number of messages metric, however this time the naive algorithm shows only moderate growth compared to the row arrangement case. This is because in a grid arrangement the shortest path takes less hop and even the naive approach is able to hit the shortest path infrequently.

In case of grid arrangement the DD and GBR algorithm shows similar performance despite the fact that GBR needs twice as much messages for operation.

Conclusion

In summary it can be stated that the naive flooding routing protocol unacceptably overloads the network even in the simplest experimental case when all the nodes are arranged in a row.

The simulation results show that GBR and DD have a superior performance over flooding in either topology, and that GBR and DD have similar efficiency in a smart metering application.

Although DD has a performance advantage as the number of nodes increases, this benefit is not really appealing with a node count of an average smart meter network, as in reality such a network consists of fewer than 100 nodes.

Given the fact, that GBR is easier to implement it is the recommended solution for routing in a smart metering network.

The research is co-financed by the National Research, Development and Innovation Fund in Hungary.

Project number: PIAC_13-1-2013-0211

SZÉCHENYI 2020

MAGYARORSZÁG KORMÁNYA

NEMZETI KUTATÁSI, FEJLESZTÉSI ÉS INNOVÁCIÓS ALAP

BEFEKTETÉS A JÖVŐBE

References:

J. Al-Karaki and A. E. Kamal, “Routing techniques in wireless sensor networks: A survey,” IEEE Trans. Wireless Commun., vol. 11, pp. 6–28, Dec. 2004.

Arampatzis, Th.; Lygeros, J.; Manesis, S., "A Survey of Applications of Wireless Sensors and Wireless Sensor Networks," Intelligent Control, 2005. Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation , vol., no., pp.719,724, 27-29 June 2005

S. M. Abd El-kader, “ Performance Evaluation for flat and hierarchical WSN routing protocols”, The Mediterranean Journal of Computers and Networks, Vol. 7, No. 3, 2011

Hongseok Yoo, Moonjoo Shim, Dongkyun Kim "A scalable multi-sink gradient-based routing protocol for traffic load balancing", EURASIP Journal on Wireless Communications and Networking, 2011