Department of Electrical Engineering, Mathematics and Computer Science

Partition-based Network Load Balanced Routing in Large Scale Multi-sink Wireless Sensor Networks

Master Thesis

Thijs Mutter
0100439

Supervisors
dr. ing. P.J.M. Havinga
dr. ir. L.F.W. Van Hoesel
MSc. A. Erman-Tüysüz

Partition-based Network Load Balanced Routing in Large Scale Multi-sink Wireless Sensor Networks

This thesis is submitted in partial fulfilment of the requirements for the degree of Master of Science of the programme in Computer Science

Author
Thijs Mutter
0100439

Supervisors
dr. ing. P.J.M. Havinga
dr. ir. L.F.W. Van Hoesel
MSc. A. Erman-Tüysüz

University
University of Twente,
Department of Electrical Engineering, Mathematics and Computer Science

Printed
January 5, 2009

Abstract

Traditional wireless networks form the wireless counterpart of wired networks, in which providing infrastructure is the main functionality. High bandwidth is the primary goal and the unlimited power supply is an important characteristic of traditional wireless networks. On the other hand, *Wireless Sensor Networks* (WSNs) are used for environmental monitoring under, sometimes, harsh environmental conditions. Their focus does not lie on providing high bandwidth, but achieving low energy consumption as well as autonomous functioning and self-deployment. The sensor nodes of a WSN are inexpensive devices, with low memory and processing capabilities and a low bandwidth. It is often costly or impossible to replace batteries and therefore WSNs need to run autonomously for many years on a limited energy source. Data, in the form of environmental sensor readings, is sent from sensor nodes to the data sink – also named gateway. The sink forms the gateway between the WSN and the end-user application. These sink nodes have more capabilities than normal sensor nodes, i.e. they can communicate directly with each other via a high-speed link, have more processing power, and are powered by an unlimited energy source. The final destination of all sensor data generated in the sensor nodes is the data sinks in the network. In some situations the application demands more than one sink in the network, in other situations a multi-sink network is created as the result of merging two single-sink networks. In all situations it has certain benefits to add additional sinks to the network, although they can easily turn into drawbacks if the routing protocol is not suited for multi-sink networks.

The aim of the research set out in this thesis, is to develop an efficient routing protocol which utilizes the existence of multiple sinks in the network. Therefore this thesis presents the *Partition-Based Network Load Balanced routing protocol* (P-NLB), a novel routing protocol for routing in large scale multi-sink WSNs. The protocol is part of the network layer in the OSI layer model and extensively uses the cross-layer from the MAC protocol in the data link layer. Sensor nodes use this cross-layer information to obtain a local view of the network neighbourhood. As an application can have different targets, for example low network lifetime, low-latency or high data throughput, P-NLB is able to deal with these different targets. It uses a network wide inter-cluster load balancing technique in combination with metric-based intra-cluster shortest path routing. In this two-level approach sinks collect information from nodes in the network about cluster sizes and distribute this analyzed information back into the network. Sensor nodes use this *global* information in combination with *local* information about the one-hop network neighbourhood to build a routing tree. This routing spanning tree is used for forwarding data from nodes to the sink. This routing mode which combines global and local information is called *Load Balancing Mode* (LBM) of P-NLB and implements inter-cluster load balancing. P-NLB also features a more basic *Smart Shortest Path Mode* (S-SPM), which lacks the load balancing feature. In the setup phase of the network it is detected whether the network should function in load balancing LBM routing mode or the basic S-SPM routing mode. The network topology is the key factor in that decision. If the protocol detects that the cardinalities of the clusters in the network are not equal, it will activate the LBM routing mode in order to restore the balance. Otherwise inter-cluster load balancing is not necessary and S-SPM routing is activated. Both routing modes feature a metric-based routing tree building mechanism, which nodes use to follow a certain routing strategy i.e. avoid congested or nearly depleted nodes on the routing

path. Four *routing metrics* are defined, and simulations must make clear which routing metric is best for which application target. P-NLB is designed for large-scale networks, since nodes route their data without a centralized control. Also, the protocol does not use a network-wide broadcasting mechanism, instead nodes only use locally available information. It is very suitable for multi-sink networks, since its load balancing technique is able to spread the load uniformly over all the sinks in the network.

P-NLB has been compared to two existing routing protocols, the centralized NCLB protocol and basic shortest path routing (SPR) in extensive simulations. The results of these simulations show that the centralized NCLB protocol performs better than P-NLB in almost every case, whereas P-NLB in turn performs better than SPR. In random network topologies, the load balancing mode of P-NLB does not perform as well as expected. Reasons for this are, among others, local bottlenecks in the network, which have a greater negative impact than the more uniform load distribution can compensate for. Another reason is the fixed but limited bandwidth of LMAC, the underlying MAC protocol, due to the use of a TDMA mechanism. This causes congestion in nodes neighbouring the sinks, instead of causing congestion in the sinks themselves, which have, as being terminal stations, a higher bandwidth for further processing.

If the nodes in the network use routing metric *Buffer* – which leads to nodes avoiding nodes with high buffer occupancy – the network achieves the lowest end-to-end latency and highest packet delivery ratio. If the nodes in the network use routing metric *Network lifetime* – which favours routing to nodes which have the most remaining energy – the highest throughput and the longest network lifetime are obtained. In comparison with SPR, the performance gain by using P-NLB is up to 50% for end-to-end latency and between 5% and 20% for performance metrics packet delivery ratio, throughput and network lifetime.

Preface

Finally I'm able to show you the work I have been working quite some time on. It was not an easy task and more difficult to complete than I expected, but now I'm proud to present you the results of all my hard work on my research in this thesis. I would like to thank my supervisors Paul, Lodewijk and Aysegul, my fellow students, girlfriend, family and friends for supporting me and keeping faith in a good outcome. I would especially like to thank Aysegul who helped me with the daily work. I really enjoyed the week of long hard working days on the AWARE experiments in Utrera, Spain. Although my first experience with programming nodes and establishing networks was not always as successful as I hoped for, they were nevertheless valuable and interesting. It was also fascinating to work together with the other researchers and see how their unmanned helicopters drop a bunch of sensor nodes from the sky, in a diaper-like package. I'll also never forget how difficult and funny it is to order one vegetarian and one pork-less meal in a Spanish city where absolute no single waiter speaks English. Yes, it was definitely a tough, but fun and above all an interesting experience and I'm glad I got the opportunity to go there. When my work on this thesis was almost finished, I got another great opportunity: writing an article about my research for the ISADS '09 conference. Well, writing your first paper appeared to be quite a tricky thing to do, but fortunately my supervisors helped me a great deal with it. After finishing and correcting it has fortunately been accepted for inclusion in the proceedings of the conference.

I mentioned AWARE before since this thesis is part of research of the AWARE project. The AWARE project is European project funded by the Information Society Technologies of the European Union and the University of Twente / CTIT is one of the partners of that project. In the introduction of this thesis a more elaborate description of AWARE and a link to its website can be found.

Well, enjoy reading this thesis, and I hope you learn something from it and after reading it, be greedy to read and learn more about the interesting field of *Wireless Sensor Networks*.

Thijs Mutter

Enschede,
January 5, 2009

Table of Contents

List of Figures

List of Tables

Chapter 1

1. Introduction

1.1 An introduction to Wireless Sensor Networks

Traditional wireless networks are used as a replacement for their wired counterparts, and are used as a network infrastructure in office and home environments. Their applications demand high bandwidth and a reliable connection. The hardware used is relatively expansive and has great capabilities: a powerful transmitter, enough processing power and memory storage, and most importantly, a sufficient energy source. Nowadays, there is a whole new type of wireless networks, which have totally other goals. The opposite of these traditional wireless networks are the *Wireless Sensor Networks* (WSN). They are used for a variety of tasks, such as environmental monitoring and ambient systems.

A WSN consists of a large number of distributed *sensor nodes* that organize themselves into a multi-hop wireless network. In such a network, a sensor node is inexpensive, has low processing and memory capabilities and a very limited energy source. In Figure 1 an example of an experimental version of a sensor node is shown. Despite these poor capabilities, the nodes are supposed to last very long on their limited energy source and the network design should allow it to be self-organizing and self-healing. The biggest challenge for WSN is to run unattended for years on their limited energy source; therefore, energy efficiency is the key property of WSN.

Every sensor network has a goal: sensing data. This sensor data is of use for a certain end-user application. In most cases, this application is not directly part of the network, but it is somehow connected to the sensor data network. The connection point between the sensor network and the other end-user network is called a *data sink*, *gateway* or *point of interest*. All the data in the sensor network is collected by the sink and send to the end-user. Networks might contain multiple sinks.

There are several basic techniques to achieve an energy efficient network. One technique is by obtaining a very low *duty-cycle* – a node communicates with other nodes for a short time, after which it goes into sleep or standby mode for a longer time. An intelligent *data link layer* protocol is needed for scheduling the active periods of the nodes. Another technique for achieving energy efficiency is *clustering*. Clustering can help create a hierarchical structure in the network which makes data aggregation easier and helps to increase the maximum lifetime of the sensor network. The *routing* protocol is also an area where energy can be saved and this thesis will focus on the design of an energy efficient routing protocol for large-scale wireless sensor networks containing multiple sinks.

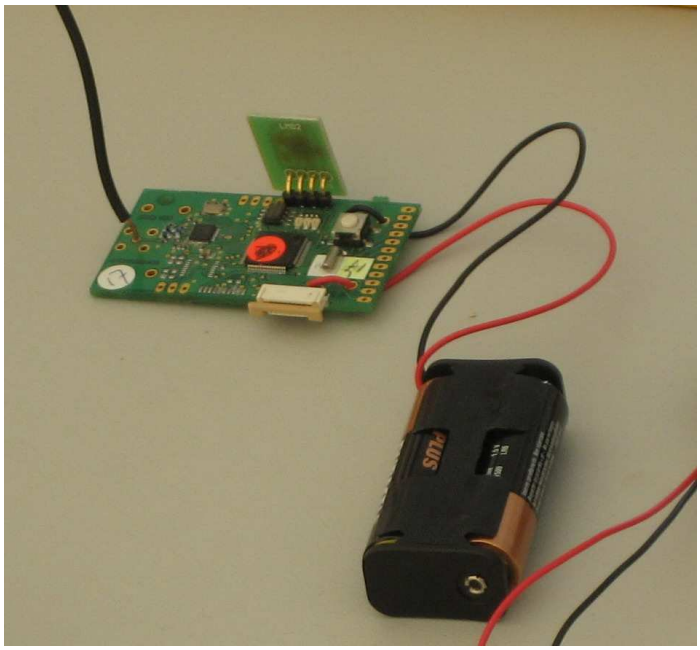


Figure 1 – Experimental version of sensor node

1.1.1 WSN applications

There are many applications where WSN prove their value. Ambient systems and environmental monitoring are two of the key applications of WSN, for example:

- Temperature monitoring in a cold store
- Humidity measurements in agricultural field
- Fire detection in an office building or forest.

WSN can also be part of a more complex platform, such as in the AWARE project [8].

Now, short descriptions of the two fire detection scenarios and the more complex AWARE project are given.

AWARE Project

The AWARE project has goal to develop a platform, which combines mobile autonomous vehicles with a ground sensor-actuator wireless network to enable the operation in difficult to reach sites, without a communication infrastructure. In this scenario a WSN is used as a fixed wireless infrastructure which can be used by various upper services for given sensor readings and passing communication messages. In addition mobile sensors attached to people make use of the WSN as wireless infrastructure without being part of that infrastructure. Unmanned helicopters are able to autonomously transport various loads. More information about ware can be found at [8].

Fire detection scenario 1: office building

Imagine a large office building in where every room and corridor has sensor nodes with smoke, temperature and humidity sensors, which is basically an advanced smoke detector. Instead of wiring them all together into a large wired network, the wireless medium is used for communication, which makes it

much more cost effective to deploy. Every sensor node will periodically sample its sensors and send a report of this to the central data sink. In this scenario the deployment, positioning and maintenance of the sensor nodes are highly controllable, although there are also scenarios possible in which this is not the case.

Fire detection scenario 2: forest

In this scenario fire detection takes place in a forest which is likely to catch fire during dry summers. In order to prevent large scale fires in the forest, it is essential to detect fire in an early stage. This forest is monitored by deploying a large number of sensor nodes to detect fire – for example, by taking temperature and/or humidity readings. The environmental condition outside a forest is much harsher than inside an office building; therefore the chance of failure of sensor nodes is much larger; thus there is the need for redundancy. There should be many sensor nodes and multiple data sinks in such a network. Node deployment is done by throwing a load of nodes out an airplane; therefore, careful positioning of nodes is not possible. Replacement of depleted or malfunctioning nodes is neither feasible. An illustration of a WSN for fire detection is shown in Figure 2. All sensor nodes – drawn as circles – periodically send temperature measurements to one of two sinks – drawn as triangles. A high temperature indicates a fire in the forest. These sensor readings are sent to one of the sinks, in several steps via multiple other nodes. A more elaborated example of this scenario can be found in [27].

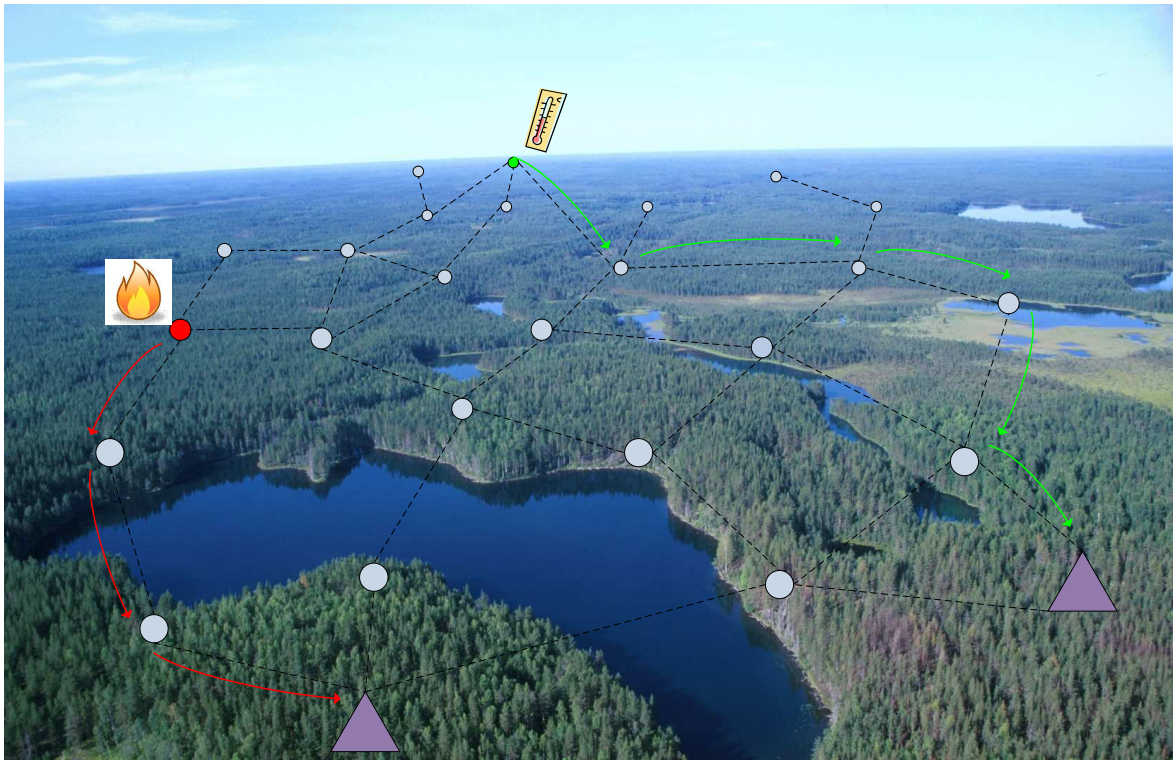


Figure 2 – Fire detection wireless sensor network example

1.2 Main contribution and organization of the document

Most of the sensor networks have one sink, but some may have multiple sinks. Having multiple sinks in the network gives great advantages and sometimes might even be necessary, but can also cause problems

if not utilized properly. Routing in multiple sink networks is not trivial if it needs to be done in an (energy) efficient manner. As the *Related Work* chapter will show many research efforts have been performed on the field of multi-sink routing protocols; however, none of them are able to meet the requirements of the networks under consideration of this thesis. These requirements will be described in the next chapter. This thesis describes a *novel routing protocol for efficient routing in large-scale multi-sink wireless sensor networks*. The new protocol is designed to efficiently utilize the existence of these multiple sinks in the network. It basically combines a network wide clustering technique with local routing optimizations, which makes it on both global and local level energy efficient.

This document will continue with a clear definition of the problem definition in the next section. In Chapter 2, an overview of existing relevant related work is presented. Next, Chapter 3 describes in great detail all key features of the novel routing protocol. Chapter 4 describes the simulation setup and the results of simulations of the new protocol compared with one related work. This thesis ends with a conclusion on the results and a discussion of future work in Chapter 5.

1.3 Research question: packet routing in multi-sink WSN

1.3.1 Definition of a multi-sink wireless sensor network

WSNs come in numerous different types, but they all share some common properties. The network nodes are small devices with very limited capabilities, i.e. few processing power, memory capacity and a finite (small) amount of energy. The WSN is a multi-hop mesh network in which not all nodes can communicate directly with each other, due to the limited transmission range of nodes, but they transmit their data via multiple other nodes to each other. Due to their limited energy source and the need of a long network lifetime –in the range of a few years – it is very important to ensure very low power consumption per node.

The particular network type we consider in this thesis consists of many nodes – varying from fifty to a few hundred. We assume that the communication within the initial network forms a connected graph i.e. all nodes can communicate directly or via multiple other nodes with each other. In the network are a few data sinks available which are different from the other nodes. These sink nodes have more capabilities than normal sensor nodes, i.e. they can communicate directly with each other via a high-speed link, more processing power, powered by an unlimited energy source. The final destination of all sensor data generated in the sensor nodes are the data sinks in the network. Transportation of data from data sinks to the end-user application is not covered in this thesis.

An example of an uninitialized multi-sink network is given in Figure 3. In the next part of this chapter this example is used to show some of the benefits of having a multi-sink network compared to a single-sink network.

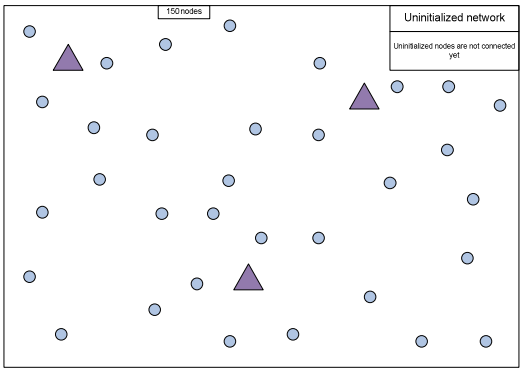


Figure 3 – Uninitialized multi-sink network

1.3.2 The pros and cons of multi-sinks networks

In some situations the application demands more than one sink in the network, in some other situations a multi-sink network is created as the result of merging two single-sink networks. In all situations it has certain benefits to add additional sinks to the network. We discuss a few of them:

- Reducing scalability problems
- Adding redundancy
- Mobility

Reducing scalability problems

Reducing scalability problems is one of the main reasons to have a multiple sink network. Scalability in networks means that for good scalable networks the size of the networks, most times expressed as the number of nodes it contains, has no (great) influence on the performance of these networks. On the other hand, a network, which badly scales, will suffer from severe performance losses when more nodes are added to the network. Problems with increasing the scale of the network can be expressed in the following terms:

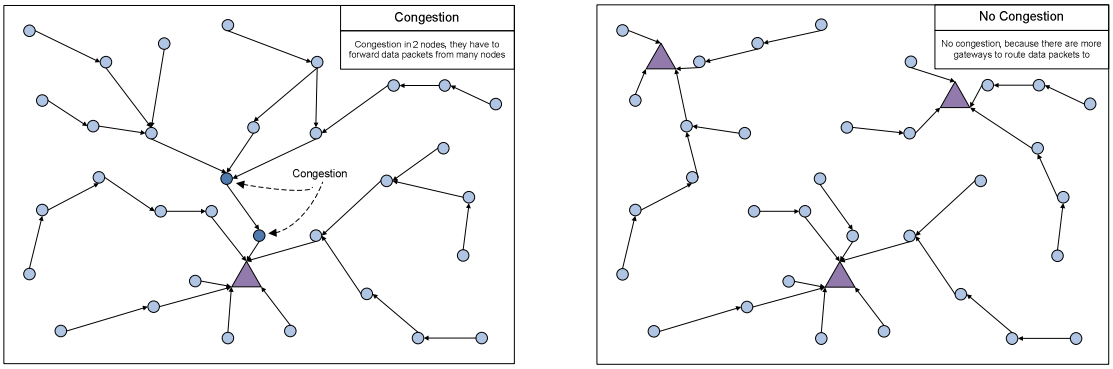
- **Increased routing path length:**
  If the deployment area of the network does not increase, only the density of the nodes in the network increases when adding more nodes to the network. However in many cases the area of deployment also increases, which results in longer path lengths from nodes at the network border to the sink. Adding extra sinks to the network causes the average routing path length of a node to decrease, because the geographical distance between nodes and sinks is smaller. Therefore the amount of hops a packet has to travel to reach a sink is smaller. As each travelled hop means the packets consumed energy at the visiting node, travelling fewer hops results in less energy consumption. The packet latency also benefits from a shorter path length, since each travelled hop causes the packet to reside some time in the packet buffer of the visiting node.
- **Congestion and load balancing problem:**
  Every node has a certain processing capability – the amount of data packets it can receive and forward during a certain period time. If a certain node receives too much data from its neighbouring nodes, it cannot forward all this data fast enough since the packet buffer of the nodes fills up until it is completely full. This is called *congestion* and as a result the forwarding of the packets is delayed or the packets might even be lost. In some networks the total traffic load is not completely overloading the network, but due to inefficient routing the load is just concentrated on one point in the network, causing congestion at that point. This problem of an unevenly distribution of traffic load is called a *load balancing problem* and is applicable to both single- and multi-sink networks.

In a small single-sink network, the sink and nodes around this sink will be able to process the amount of traffic. However, if the network grows beyond a certain size, the amount of traffic, coming from all around the network, will be too much for the sink and nodes to handle. As a result congestion will occur in the area around the sink and in this case adding extra sinks to the network can solve this scalability originated congestion problem. In scenarios where the placement of these (extra) sinks can be controlled, sink placement and clustering algorithms have been developed [20], [19]. These algorithms are designed for optimal placement of the sinks in the network in such way that the total network load is uniformly distributed over all sinks. However, in many scenarios the deployment of the sensor nodes and data sinks and hence the network structure cannot be controlled. In such an unpredictable *random* network structure nodes and sinks are not uniformly distributed over the area. As a result, sinks can be grouped closely together or many nodes are grouped around a small part of the available sinks. The load of this large group of nodes leads to congestion at this small number of sinks, while the other sinks have much processing capabilities left. This is another instance of the load balancing problem, but now in a multi-sink network. As research shows in [22], multi-sink networks benefit from clustering in order to balance the load in a network uniformly over the sinks in the network. The approach in [2] describes the effects of energy depletion and reliability on the connectivity of a WSN. They conclude that nodes close around a sink have a higher chance of failing than nodes with a higher hop count, because nodes close around a sink have more traffic to process. For those mentioned reasons a load balancing approach has several benefits:

- It prevents congestion around one sink, while other sinks have no traffic to process, therefore increasing the throughput and decreasing the latency around the sinks.
- It prevents energy depletion in the nodes around one sink, while nodes around other sinks have a large energy reserve.

In Figure 4 a) and b) is shown how congestion can be prevented in a single-sink network by adding two extra sinks.

- **Communication overhead:**
  Depending on the routing protocol, large networks will suffer from more and more communication overhead. Sinks needs information about nodes in the network and send command and status information into the network. A simple but inefficient broadcasting protocol causes an exponential increase of communications.



a) Congestion in network due to limited amount of sinks b) Adding sinks eliminates congestion

Figure 4 – Congestion due to single sink network

Adding redundancy

Having only one sink in the network causes problems in case of failure of the sink or nodes around it. In that case the data in the network has no route to reach the end-user application, which makes the whole WSN completely useless. Multi-sink network are therefore more resilient for node failures.

Mobility

In case of mobility having multiple sinks in the network might be a must. Groups of nodes in the network may move in the network and might move out of range of the rest of the network, but when this group contains a sink it will not be disconnected from the network. An example of such a network can be found in a storage facility. Such a facility has a static network infrastructure, including one or more sinks. On the shelves in the facility are boxes attached with sensor nodes periodically sending temperature readings to the sinks. When a shipment of boxes is loaded into a truck, which also contains a sink, the network splits into two clusters, but the nodes in the truck become not disconnected from the network.

Multi-sink Evaluation

Adding extra sinks to a network helps reducing scalability problems, but without a clever designed routing protocol that limits the extra communication overhead and load balancing problem, a multi-sink network might not have any better performance or even be outperformed by a single-sink situation. A limited budget is also one important reason to limit the amount of data sinks in the network to a minimum, because due to their enhanced capabilities, the costs of a data sink is in general much higher than that of a common sensor node.

The next paragraph describes the research question of this thesis and the demands on the new routing protocol in order to overcome the utilization problem of multiple sinks and at the same time benefit from having them.

1.3.3 Thesis goals

The goal of this thesis is developing an efficient routing protocol which utilizes the existence of multiple sinks in the network. As an application can have different targets, for example low network lifetime, low-latency or high data throughput, the new routing protocol must be able to deal with these different targets. In Chapter 3.4.1 this subject of application demands is further explained.

The summary of the aims of this thesis is as follows:

- Utilizing the advantages of having multiple sinks in the network and at the same time avoiding problems caused by having multiple sinks. A load balancing algorithm might be useful for this. More information about load balancing is given in the Related Work section and in Chapter 3.
- Designing a more efficient routing than random shortest path routing algorithm. Depending on the application the new protocol must be able to achieve high network lifetime, low latency or high throughput.
- Solution scalable to large-scale networks with many sensor nodes and data sinks.
- No need for geographical location information.

In order to verify if these goals are met, simulation must be done with some performance measurements. There are several *performance evaluation metrics*. Different applications have different demands on the network, and these demands can be benchmarked with different metrics. Some common metrics will be discussed in Section 1.5.

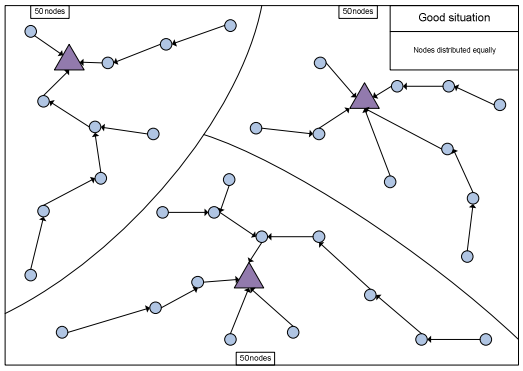


Figure 5 – Goal: balanced multi-sink network

1.4 System description and assumptions

This paragraph gives a detailed description of the system model and all of its components and some assumptions to limit the subject covered by this thesis. This description is given at this point because many of the terms explained here are used in the rest of this thesis.

These following terms are important for understanding the operation of a wireless sensor network. Some of them are illustrated in Figure 6. The network self is a representation of a graph *G*, with vertices *V* as nodes (sensor nodes and data sinks) and edges *E* as communication links. *N* is the number sensor nodes, *M* the number of data sinks, $N \subset V$ and $M \subset V$ and $N \gg M$

- **Sensor node**. Low processing and memory capabilities, limited power supply.
- **Data sink**. More capabilities than common sensor nodes: more processing power, unlimited power supply. Connected to end-user application (network). Sometimes
- **Communication link**. Bidirectional link between two sensor nodes, which can be used for exchanging information. There is a communication link between a pair of nodes if they are within transmission range of each other. Sometimes abbreviated to *link*.
- **Neighbour**. Two nodes are neighbours of each other if there is a communication link between those two nodes.
- **Hop count**. The hop count is the shortest distance between a node and a sink, measured in hops. A packet travels one hop if it travels from one node to its neighbour. A packet travels two hops if it travels to a node via another node.
- **Child** and **Parent** nodes. Each sensor node has a vector pointing to a neighbour node, representing to which neighbour a data packet is forwarded. The sending node is the child node; the receiving node is the parent node.
- **Spanning tree**. All vectors form one spanning tree in the network. In case of multiple sinks, multiple spanning trees are formed. All nodes of a spanning tree form a *Cluster*.
- **Routing path**. Path which packets use to travel from source node to the data sink.
- **Descendants**. The descendant nodes – sometime called upstream nodes – of a node are the nodes that are on the same routing path, but have a higher hop count – in other words are further away – from that specific node.
- **Branch**. A sink has one or more neighbours; these neighbours are called *top-level nodes*. These nodes are the beginning –the root – of a *top-level branch* (sometimes called just branch).

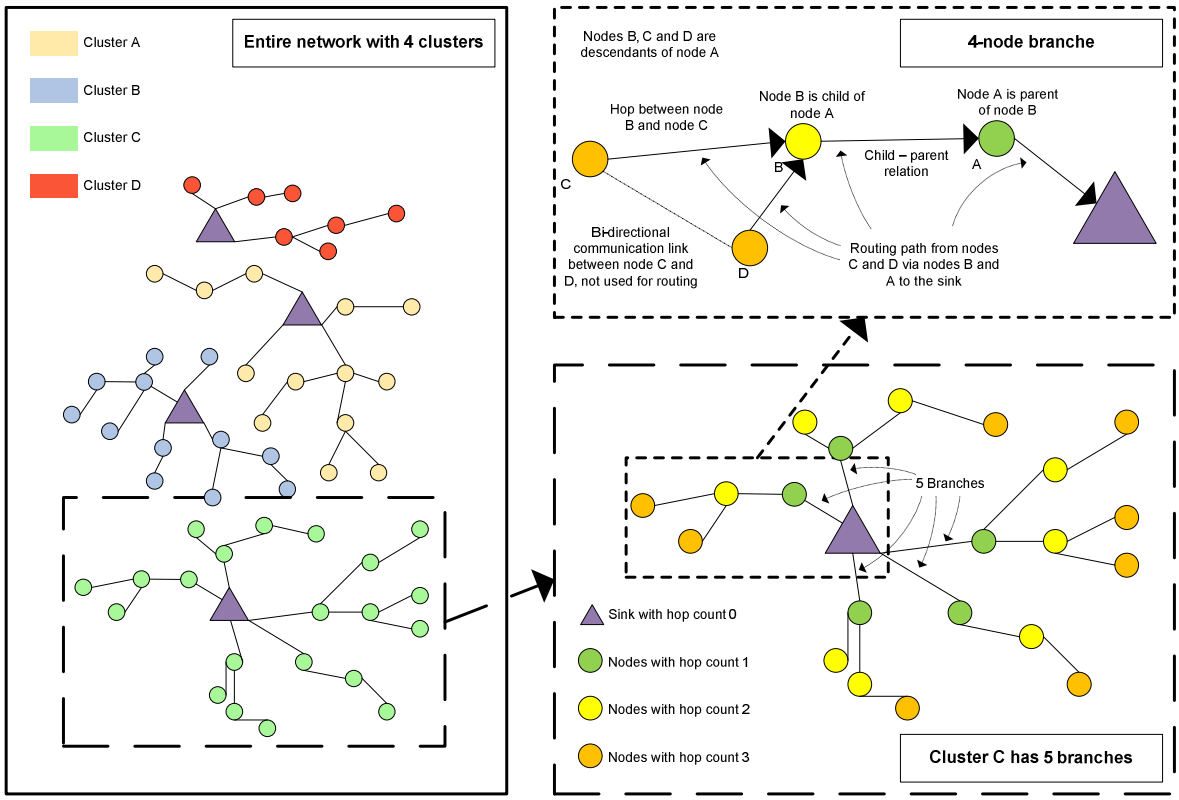


Figure 6 – Wireless sensor network components

Assumptions

A WSN has many facets and parameters and in order to able to control the complexity of the network model in this thesis, some assumptions about the network model are made:

- Sinks and sensor nodes in the network are stationary; they don't change their position.
- Ideal circle shaped stable radio transmission medium: only bi-directional communication links between pairs of nodes and communication links between nodes don't change over time. The radius of the transmission range of nodes is much smaller than the size of the area where nodes are deployed, therefore direct communication between all nodes in the network is not possible.
- There is only one pattern of dataflow in the network: from sensor nodes to data sinks. The data sinks will not send packets to specific sensor nodes in the network. This is the most common communication paradigm in (data gathering) sensor networks [1].
- Sinks can (directly) communicate with each other using a high-speed communication channel.
- Sinks are equal from the information point of view; it doesn't matter to which sink a data packet is send. We assume that after reception of the packets all sinks forward them to the same end-user application.
- There is useful cross-layer information exchange between data-link and network layer i.e. information about neighbouring nodes.
- No data aggregation is done by nodes in the network.
- All nodes in the network generate the same amount of traffic. This is a common situation in WSN designed for environmental monitoring, where data packets only consist of fixed size sensor readings.

1.5 Performance evaluation metrics

In order to test the performance of the new routing protocol this thesis defines a set of performance evaluation metrics. There are different evaluation metrics, because there are also different demands on the network, depending on the application. More about application demands can be found in Section 3.4.1. The simulation results will be analysed with use of these metrics in Chapter 4.

Latency

Latency – sometimes called *end-to-end delay* – is an important factor in for many WSN application types. In this thesis we are primarily concerned about the *downlink latency* – the latency between sending a packet at the source node and receiving the packet at a data sink – because in most WSN traffic flows in that direction. *Uplink latency* is the latency between sending a packet at the data sink node and receiving the packet at a sensor node.

Latency in the network has different sources:

- **The lengths of the paths from nodes to sinks** – This is affected by the structure of the spanning tree(s) in the network. If the paths between the source-sink pairs are long on average, the latency will also be higher. Every hop in the path to the sink a packet has to wait a certain time, so more hops in this path will directly lead to higher packet latency. Depending on chosen metrics, the average path length in the network varies, so does the latency.
- **The timeslot latency between every – one-hop – pair of source and destination nodes** – On the data link layer there are several different medium access techniques. One of them is a *slotted reservation based* medium access technique where the time domain in divided into *timeslots*, each of a predefined time. Each node *reserves* one timeslot, which it uses for transmission. In theory one timeslot is reserved by only one node, resulting in only one transmitting node at any time, thereby eliminating collision on the wireless medium. A packet travelling from source node *A* via intermediate node *B* to destination node *C* is confronted with a delay at intermediate node *B*. At a certain moment node *A* sends the packet in its timeslot to node *B*, but node *B* cannot forward the packet to node *C* until the reserved timeslot of node *B* comes. This delay between the receiving timeslot and forwarding timeslot is called the timeslot latency and has its source in the MAC protocol in the data link layer.
- **The buffer occupation of the nodes in the network** – The last source of latency is caused by the time a packet resides in transmission queue of a node, before it can be transmitted. When a certain packet *x* arrives at an intermediate which already has 4 other packets waiting in its transmission queue, it must wait before these other packets are transmitted, before that packet *x* can be transmitted. Assuming a node can send one packet every reserved slot, packet *x* suffers from an extra delay equal to the time between two reserved timeslots – called frame duration – for every packet in front of him in the transmission queue.

By taking these sources for latency in consideration and assuming a node can send one packet every frame, the estimated average packet latency is:

$$avg.latency = avg.source\text{-}sink\ \ pathlength * avg.timeslot\ latency + avg.buffer\ occupancy \quad (1.1)$$

When a routing protocol uses best effort routing, packets are dropped on transmission errors instead of resend. This has an effect on the latency, because these dropped packets are not taken in account with the latency measurement, only packets that arrive at a sink are included in the measurement for that.

For some application the standard deviation of the latency or the maximum latency might be more important than the average latency.

Network lifetime

The network lifetime is important for networks where nodes have a limited energy source. The replacement of these energy sources is very costly, or even impossible. Sensor nodes in *idle mode* have certain basic energy consumption, caused by basic activity of node components and periodically short transceiver activity. In the *active mode* the transceiver is switched on for a much longer period, resulting in much higher energy consumption [7]. This active mode is necessary for forwarding data packets and thus forwarding data packets costs a lot of extra energy. Therefore reducing the amount of packets nodes have to forward saves a lot of energy. Using our approach of spanning trees with the sinks as roots has certain consequences for the energy dissipation of the nodes close to the sinks. In [2] a study has been conducted which researches the effect of the failure rate of nodes. Concrete this means that nodes in close proximity of the sinks become depleted than nodes at the border of the network. By extending the lifetime of single nodes, the lifetime of the whole network can be extended. The lifetime of a single node can be extended by smartly monitoring the energy level of the node and adapting the amount of traffic the node has to handle, i.e. rerouting traffic around nodes with low energy levels towards nodes with higher energy levels. This can be achieved by using intelligent routing methods and this performance metric measures the effectiveness of these routing methods.

There is no unified definition of the network lifetime, since this concept depends on the objective of an application. Instead, several definitions of network lifetime can be found in the literature [4, 5, 23, 27]:

- Time from initialization until the first node fails.
- Time from initialization until the first network partition occurs.
- Time from initialization until a certain percentage of nodes fails.
- Time from initialization until the last node fails.
- Time from initialization until a certain percentage of coverage remains.

In this thesis the first definition is used. There can be several reasons for nodes to fail; physical failure – where a certain part of the node fails due to mechanical failure – or the energy source of the node becomes depleted. Although there is a certain chance of mechanical failure, the main cause of node failure is energy depletion.

Throughput

Throughput is the amount of data a network processes (per instance of time, frame length for example). It is measured as the number of packet arriving at the sinks during one frame. The packet rate – the rate at which nodes send their sensor readings to the sink, in other words “generate” data packets – has a great influence on the throughput. A high packet rate leads to many packets being generated in the network and arriving at the sinks. It is also influenced by the number of top-level branches of the sinks, when using the LMAC protocol. Due to the TDMA mechanism of LMAC, every node can send only one packet each frame, consequently, a sink with x neighbours can only receive a maximum of x packets every frame.

The network throughput at a certain point of time can be defined as the amount of packets received at all data sinks during that certain period of time. We will measure the average throughput of the network, which can be represented by the following equation, β is the simulation duration in frames and ε is the number of data sinks in the network:

$$throughput = \frac{\sum_{i=1}^{\beta}\left(\sum_{j=1}^{\varepsilon}\text{received packets in frame } i\right)}{\beta} \tag{1.2}$$

Energy efficiency

Since nodes in a WSN have a finite energy source, *energy efficiency* is another important issue. The energy efficiency of a routing protocol shows us how efficient the protocol is utilizing the available energy in the network in order to deliver as many as packets possible at the data sinks. Although it cannot be controlled, the hop count has quite an influence on amount of used energy for packet delivery; however, the length of the routing path can be controlled.

Protocols which have much communication overhead or higher packet loss might not be very energy efficient, because a packet that is lost halfway on its path wastes all the energy to get that far. The communication overhead of a routing protocol is considered as all the communication between nodes for setting up the initial network and maintaining the routing paths in the network, without actually transporting data packets.

The energy efficiency of the network can be measured by using many different definitions. In this thesis the energy efficiency is measured as the *energy-per-message* (EPM): the total energy used in the network divided by the number of successfully delivered packets at the data sinks. The total energy used in the network includes the energy used by the other protocols in the stack, such as the MAC protocol, but when different routing protocols use the same MAC protocol this influence should be negligible. The energy used by a node's electronics is also taken into account. The energy-per-message can be written as the following equation:

$$epm = total\ energy\ used\ by\ all\ nodes\ /\ all\ \ packets\ delivered\ at\ all\ sinks \tag{1.3}$$

Packet Delivery Ratio

The Packet Delivery Ratio (PDR) gives a lot of information about the efficiency and reliability of the network. WSN have several sources of packet loss, where transmission failure due to the unreliable wireless medium is the main source. Buffer overflows due to full buffers are another common cause for packet loss, especially in case of best effort routing. Due to various reasons sensor nodes can also temporarily or permanent fail, which causes the packets which resides in the nodes buffer to be lost.

$$PDR = number\ of\ \ all\ \ packets\ received\ at\ sinks\ /\ number\ of\ \ all\ \ packets\ generated\ at\ sensor\ nodes \tag{1.4}$$

Another subtle issue about the PDR is fairness, by which is meant that nodes further away from the data sinks are likely to have a lower PDR that nodes closer to a sink. Therefore, the fairness is the PDR as function of the hop count.

Load per sink

Although not a goal itself, it is useful to test the how effective the load balancing algorithm balances the load in the network. A balanced network is not a goal by itself, because a balanced network doesn't guarantee a good performance, it could, for example, still have a higher latency or lower throughput compared with an unbalanced network. The average load of all sinks is just another term for the throughput as defined earlier, but what is more interesting in this case, is the variation of the load of the sinks in the network. This gives a good indication for the distribution of the load in the network. For that reason, we will measure the load per sink performance metric as the standard deviation of the load on each the sinks in the network.

The standard deviation of the sink load, σ, is given in the following formula, where M is the number of sink, l, the average load of all sinks and $\bar{l}$ the difference between l and the load in sink i.

$$\sigma = \sqrt{\frac{1}{M} * \sum_{i=1}^{M} \left(l_i - \bar{l} \right)^2} \qquad (1.5)$$

Chapter 2

2. Related Work

The field of WSNs has a relative short history. In 2001, *Power Aware Clustered TDMA* (PACT) [3], one of the first protocols with special MAC and clustering techniques for WSN, was developed. Since then, much research has been done on the several network issues where WSN differ from traditional networks.

2.1 OSI layer model

Network communication is modelled into a layered system model by the *Open Systems Interconnection* (OSI). In the OSI layer model, the whole communication protocol – from the physical medium to the end-user application - is divided into several layers. In Figure 7 all the layers of the OSI model are shown. In this thesis, we are mainly concerned with the *Data link* and *Network* layers. The network layer contains routing and *Quality of Service* (QoS) functions. The Data link layer contains the Medium Access Control (MAC) and Logical Link Control (LLC) functions. The Light-weighted Medium Access Control (LMAC) [7] protocol is used as underlying MAC protocol in this thesis. Since these layers lay close together, LMAC in the data link layer is able to pass useful *cross layer information* to the Network layer.

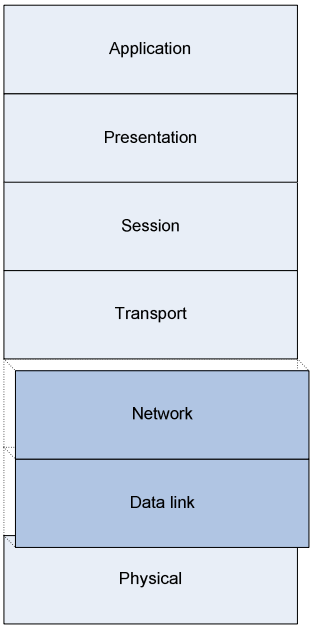


Figure 7 – OSI layer model

2.1.1 Data link layer

The data link layer primary functions are channel access control in the MAC sub-layer and multiplexing transmitted over the MAC layer and providing error and flow control in the LLC sub-layer. The link- and network-layers are very important for WSN. A large part of the scarcely available energy of network nodes is used by the transceiver. Intelligent link-layer protocols can considerably reduce this energy consumption. The link-layer protocols can be used to reduce other forms of energy waste, such as idle listening and collisions. By using intelligently designed cross-layered data link- and network-layer protocols large energy savings can be realized.

WSN data link-layer protocols differ from traditional link-layer protocols such as 802.11a/b/g due to the need for low energy consumption. Sources of energy waste are collisions and long duty-cycles. All existing link-layer protocols try to reduce these problems by using different techniques. A drawback of most of these techniques is an increase in latency and an increasing amount of communication and protocol overhead.

2.1.2 LMAC

The MAC protocol is of great importance in this thesis, since cross-layer information of the MAC is extensively used in the routing protocol. Therefore, the MAC protocol used in this thesis – LMAC – is first described. Although LMAC is used as MAC protocol in this thesis, every other MAC protocol which provides the same cross-layer information to the network layer can be used.

In LMAC time is divided into frames, which consists of timeslots, where each timeslot can be controlled by only one network node. A time slot is divided into two parts, a control and data section, where a node always broadcasts the control section to all its neighbours, which contain information about the node and its 1-hop neighbourhood. It also addresses a neighbour in this control section if it has data to send and this neighbour node will then also remain active during the data section of that timeslot. When a

node is not addressed during a timeslot it can turn off its radio during the following data section. It uses virtual clustering to avoid colliding timeslots within two hops from a node.

LMAC is a scheduled based protocol using a combination of *Time Division Multiple Access* (TDMA) and *Space Division Multiple Access* (SDMA) techniques. It has been designed to work very energy efficient in WSN. It functions without a central manager; nodes function autonomously.

TDMA scheme

LMAC is a link layer protocol using TDMA schemes for communication. Time is divided in *timeslots*, grouped together into frames, consisting of 32 timeslots. Every node can use one timeslot per frame to transmit data to other nodes. The major advantage of such a TDMA scheme above contention based schemes is the lack of collisions. Only one node will transmit during a timeslot in a frame, so no collisions, which are a source of energy wasting, occur. There are many more nodes than there are timeslots in a frame, but due to their limited transmission range and intelligent choosing of timeslots, multiple nodes can transmit at the same time, without causing interference.

Frame overview

LMAC divides a timeslot into two parts, a Control Message (CM) section and a Data Message (DM) section. The CM section contains the ID of the node, the timeslot it occupies and if the node has data for another node, it addresses this node. Every node broadcast this CM section to all its neighbours. The DM section is used for sending data; in the CM section a nodes has addressed the other node, and in the DM section it transmits the data. The addressed node received the CM section of the transmitting node and knows the data is for him and listens to the data in the DM section. Other nodes that are not addressed by the transmitting node can switch off their transmitters, thereby saving energy.

The TDMA structure of LMAC consists of 2 parts, the CM and DM sections. The sizes of the CM and DM sections are respectively 114 and 2040 bits. As already mentioned, by dividing the whole timeslot into two parts where a node must only listen to one short part – the CM section – nodes can save considerable amounts of energy by switching of their receivers.

The CM section consists of the following fields:

- Node identification
- Current occupied slot
- Distance to sink
- Occupied slots
- Collision in slot
- Parent
- Sink which forms the root of the spanning tree the node is in – in order words, the cluster
- Routing path length
- And depending on the routing metric, on of the following information:
    - Number of child nodes
    - Estimation of the number of descendant nodes
    - Buffer occupancy
    - Energy level

The DM section contains the data the node has to send.

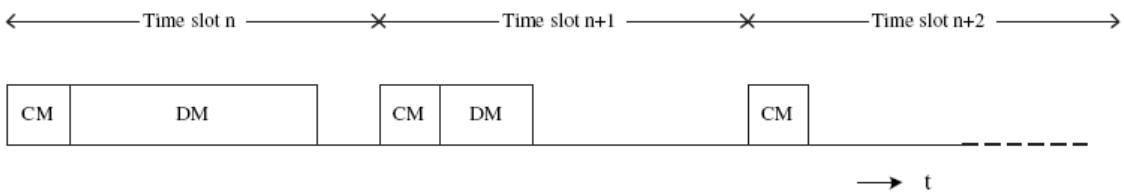


Figure 8 – Frame overview of LMAC (taken from [7])

Energy consumption reducing techniques

LMAC use many techniques which reduce the amount of energy needed for communication. This makes it very suitable for WSN. The most important one is the TDMA structure, which prevents colliding transmissions which are a common source of energy wasting, especially in dense networks. Another important technique is the addressing mode in the CM section, after which the non addressed nodes can switch off their energy consuming receivers. Synchronization of time is inherent available in the protocol, so no additional synchronization techniques need to be implemented.

2.1.3 Network layer

The network layer performs network routing functions and QoS requested by the transport layer. The network layer is responsible for end-to-end packet delivery, whereas the link layer is only responsible for node-to-node frame delivery on the same link. Routing is the task of finding a path from source to destination. The term QoS refers to the ability to priorities to different data flows or to guarantee a certain level of performance to such a data flow. QoS is not covered in this thesis.

There are two different routing mechanisms: proactive and reactive routing. A proactive routing protocol maintains a routing path and periodically puts effort in maintaining it. Every node maintains one or more routing tables for storing information about routes between nodes in the network. Topology changes are propagated throughout the network and nodes attempt to maintain consistent up-to-date routing information from each node. On the other hand, a reactive routing protocol, calculates a route only when it has data to transmit. This approach has no periodically maintenance costs, but increases the cost of finding a correct routing path if needed. The route discovery address can be source-initiated or destination-initiated. Based on the underlying network structure, WSN can be flat or hierarchical. In a flat network all nodes perform the same function. In a hierarchical network structure, some nodes have the role of cluster head, maintaining the cluster, aggregating data from common nodes and forwarding data to sink node(s).

2.2 Related work overview

In Table 1 an overview of all related works can be found with their properties related to the problem definition. All related works are part of the network layer. The protocols are discussed in the remaining of this chapter.

Table 1 – Overview of related work

Protocol	*Protocol type*	*Objective*	*Explicit Multi-sink support*
[13]	Network design	Sink placement	Yes
[20]	Network design	Sink placement	Yes
FROMS [24]	Routing	Multi-sink	Yes
[12]	Routing	Multi-sink	Yes
[11]	Routing	Multi-sink	Yes
[10]	Routing	Multi-sink	Yes
[9]	Routing	Multi-sink	Yes
ART [23]	Routing	Parent selection	No
LT [6]	Routing	Parent selection	No
LBC [22]	Clustering	Load balancing	Yes
DLBR [30]	Routing	Load balancing	No
NCLB [31]	Routing	Load balancing	No
LBSP [32]	Routing	Load balancing	No
e3D [33]	Routing	Load balancing	No
OFFIS [34]	Routing	Load balancing	No
LEACH [16]	Clustering	Load balancing	No
GLBCA [21]	Clustering	Load balancing	Yes
Arbutus [15]	Routing	Load balancing	No
RTLD [14]	Routing	Load balancing	No

2.2.1 Multi-sink routing

Most protocols described in this chapter, have not specifically been designed for use in networks containing multiple sinks. However, many researches have been done on the topic of multiple sink WSN, and WSN with one or multiple mobile sinks. There is a difference between networks with mobile sinks and static sinks. Some researches like [20] are conducted on networks which have static sinks, where the goal is to route data efficiently to these sinks. In other networks, the sinks are mobile and the goal is to find a good algorithm [13] to position these sinks in the most efficient way, in order to minimize energy dissipation at each node in the network. In our situation we only deal with a static network with static sinks and sensor nodes.

With multi-path routing a single node routes its data via multiple paths, which might contain overlapping parts, to a single or multiple sinks. Using multiple paths to route data to a single sink –or in generally, a single destination – is used to avoid packet loss due to bad links on one routing path [36, 37, 38]. However, often those paths converge at the sinks where congestion occurs in those nodes. Multi-path routing to multiple sinks might avoid this [12].

Multi-path to multiple sinks routing can also be extended to the routing problem where data from multiple sources needs to be transported to multiple sinks in an efficient way by combining parts of the different routing paths. In [11], they propose an algorithm for efficient routing in such scenarios. They first present a theoretical model of the problem for computing the theoretical optimal solution of the problem. After that, they propose a decentralized solution for the problem, based on periodically adaption of the routing trees in the network. This adaption is based on a quality metric of each neighbour, where this quality metric relies on (1) the distance from neighbour to sink, (2) the number of paths passing through the nodes and (3) the number of sinks the neighbour serves.

There are also differences between protocols where data is routed to multiple sinks [9] and protocols where data is routed to a single sink, in a *multiple-sink network*. *Feedback Routing for Optimizing Multiple Sinks* (FROMS) [24] is an example of that first type. Nodes in the network exchange local

information in order to find the best hops for forwarding the data packets. By using this technique and information from FR Framework communication overhead is minimized. Examples of information exchanged are residual node energy, available routes to sinks, link quality. The information is piggybacked on all data packets. FROMS uses a *reinforcement learning solution* to deal with the dynamic environment of the network where node failure and movement is common. Nodes incrementally learn their best next-hop on route to all destinations.

In [10] data packets are routed from one source to one sink in a multi-sink network with in addition node and sink mobility. They use geographical location information and *Received Signal Strength Indicator* (RSSI) for their opportunistic routing protocol.

2.2.2 Load balancing

In sensor networks with many nodes, some nodes might have to process more data packets than other nodes in the network, for example, nodes close to a data sink. These nodes do not only suffer more from congestion, but they also consume more energy due to receiving and transmitting data cost energy. Therefore, there are several reasons for balancing the *load* over the network nodes more uniformly, i.e. reducing congestion in nodes, extending the lifetime of the network nodes. Various techniques have been proposed in the literature which balance the load on the network: e3D, LEACH, LBC, DLBR, NCLB, LBSP, OFFIS, GLBCA, Arbutus and RTLD.

In *Low-Energy Adaptive Clustering Hierarchy* (LEACH), *Load Balanced Clustering* (LBC) and *Greedy Load-Balanced Clustering Algorithm* (GLBCA) the distribution of the load is controlled by creating *clusters* in the network containing cluster heads which gather data from the nodes within the cluster. In LBC, this data is forwarded to a single sink in the network, while the network in LEACH and GLBCA contains multiple sinks, where each sink is also a cluster head. By forming these clusters, the distance the packets in the network have to travel is reduced. In GLBCA, they define the problem of balancing the load in the clusters as *Load-Balanced Clustering Problem (LBCP)* and prove that under general conditions this is a NP-hard problem. In the special case that the load in all nodes in the network is equal, they prove that LBCP is optimally solvable in polynomial time. In *Distributed algorithm for Load Balanced Clustering* (DLBR), *Load Balanced Short Path* routing (LBSP) and *Arbutus*, the goal of distributed energy consumption is achieved by looking at the energy level of neighbours and forwarding to nodes which have a high energy level, while avoiding forwarding packets to nodes which are nearly depleted.

Arbutus focussed strongly on link quality with its build-in load balancing scheme. By accounting for network load in the route selection process, it reduces the impact of bottlenecks – called *hot spots* by the authors – on network lifetime.

The distances between each node and the distances between each node and the sink is used in *Energy Efficient Distributed Dynamic Diffusion* (e3D) as a metric for forwarding data from node to sink, directly or via multiple other nodes. In this *diffusion based* approach a node can order – via special control packets – other nodes to stop using it as a relay node if for example the message queue is full or the energy level is below a certain threshold. The proposed protocol in *Optimized Forwarding by Fuzzy Inference Systems* (OFFIS) uses a *fuzzy inference system* (FIS) [18] that optimizes the routing path in a distributed fashion. The goal of OFFIS is maximizing the network lifetime.

Real-Time routing protocol with Load Distribution (RTLD) uses *geodirectional-cast* forwarding for real-time communication in WSN. Its routing depends on optimal forwarding decisions that take into account of the link quality, packet delay time and the remaining power of next hop neighbours.

In *Node-Centric Load Balancing* (NCLB), the designers look at the structure of the routing paths from nodes to sink and use an offline method for balancing the load across different branches of the routing trees. In this offline algorithm the spanning trees are built step by step. The load at the sink is not balanced – since the algorithm assumes a single-sink network – but the load at the *top-level* nodes; the one-hop neighbours of the sink. After all, in a single-sink network the load on the sink cannot be controlled, but the load on its one-hop neighbours can. At the start of the algorithm, only the sink and its neighbours are part of the spanning tree. At each iteration the "weight" (load) of these branches and the "freedom" of these branches is calculated and lightest branch with the most freedom is expanded. They use the "Chebyshev Sum Inequality" as a load balancing metric.

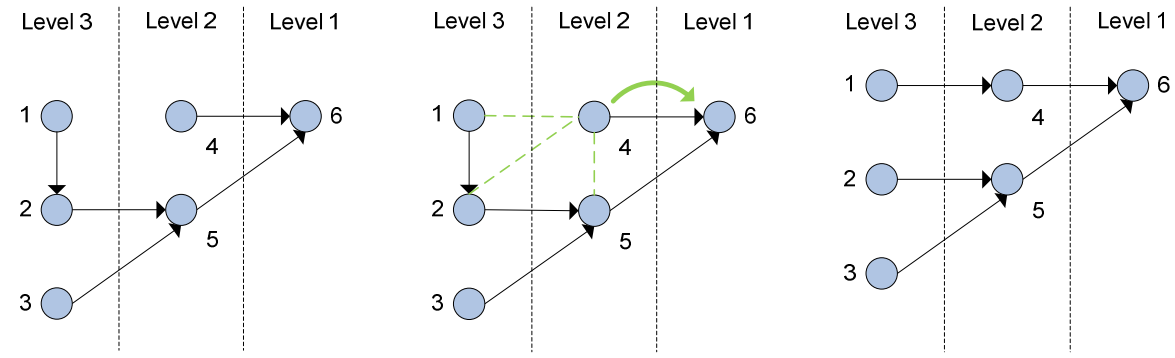
2.2.3 Parent selection

A common routing technique is building multiple spanning trees – directed acyclic graphs, in literature sometimes named routing trees – in the network, with nodes as vertices and forwarding vectors as edges.

This basic topology is desired in data-gathering wireless sensor networks, since the traffic is mainly in the form of many-to-one flows. More details about spanning trees are given in Section 3.4. Some examples of protocols which use spanning trees for routing are ART, LT, DLBR, NCLB and OFFIS. In the forwarding step, each node has to make one forwarding vector as part constructing the spanning tree, and this is called *parent selection*. Nodes can choose a new parent from one or more neighbours, based on same metrics. In *Adaptive Routing Tree* (ART), *Localized Topology* (LT) and OFFIS parent selection is an important part of the routing protocols.

In ART, one can define a cost function for each node, called *Q-value*, indicating the minimum cost-to-go from this node to the destination. A node also stores the Q-value of its neighbours, called NQ-value. Initial Q-values and NQ-values can be estimated during network initialization or upon receiving packets for the first time. During initialization phase, an initial spanning tree is built, which may not be optimal yet. Each node, other than the root node has a pointer to its parents, which is the neighbour with the smallest *NQ-value*. Nodes forward packets to their parents until the packets reach the sink. Implicit packet confirmation is used: if the packet is not heard from the node within a certain period, the node updates the node updates the NQ-value of that node and selects a new parent from the set of neighbours.

The parent selection method can be based on three different routing strategies, shortest path, energy awareness and congestions awareness. The NQ-values of the nodes depends on which strategy is used.



a) Initial routing tree

b) Node 4 sends packet to 6, nodes 1, 2 and 5 hear, in addition to 6

c) Node 1 changes its parent to 4, which has a lower Q-value

Figure 9 – Adapting spanning tree for better routing in ART

ART makes no use of multiple sinks, however, its routing concept of having different routing strategies and different routing metrics seems very effective.

In LT the network starts with a flooding initiated by the root node, followed by a parent selection phase done by all nodes, which constructs the spanning tree. They present four localized parent selection methods: earliest-first, randomized, nearest-first, and weighted-randomized parent selection.

The earlier discussed protocol OFFIS uses fuzzy logic in order to combine four different routing metrics – distance from neighbour node, distance from shortest path, remaining battery life and link usage – which leads to one parent select decision.

2.2.4 Related Work Evaluation

The related works show some useful ideas for the new to be developed routing protocol. Multi-sink routing can be done using multipath to a multiple sinks – such as in [12, 11] – in order to increase the change of successful packet delivery. As a drawback this increases the delivery costs, since in fact multiple packets are send while only one packet has to reach its destination. Depending on the network situation the extra cost are higher than the increase in performance. Since the network in consideration in this thesis assumes all sinks are equal from an information point of view, [9] and FROMS are unnecessary. The requirement of geographical information in [10] makes this solution not feasible for the network model in this thesis.

Load balancing shows promising results in single sink networks and it can be expected that it performs even better in multi-sink networks, where the load can be distributed over more sinks. All the related works have major drawbacks, which make them not suitable for the network model under consideration in this thesis. None of the related works are distributed solutions which can combine routing with load balancing and none of them uses cross-layer information for reducing communication costs.

- **Centralized.** The solutions in LBC, NCLB and GLBCA are centralized and therefore not very scalable to large networks and not very flexible in the case of changing network conditions or topology.
- **Localisation information.** LBC, OFFIS, LBSP, GLBCA and RTLD assume the availability of information about the location of the nodes in the network and therefore the need for GPS equipment or a localization algorithm on the sensor nodes.
- **Direct communication.** The e3D algorithm assumes the possibility of direct communication between all nodes in the network, while this is not always the case, especially in larger networks which require multi-hop communication model instead. LBC and GLBCA are clustering algorithms, which cluster nodes around sinks, without taking in consideration the connectivity between those nodes and sinks and the possibility of multi-hop routing.
- **Lack of multi-sink support.** Except for LBC, LEACH and GLBCA all algorithms are designed for networks with only one data sink.

Table 2 summarizes the properties of the load balancing protocols.

Optimized parent selection as opposed to randomized parent selection, as for example used in ART, seems also useful since it makes it possible to avoid local bottlenecks – in the form of congested and nearly depleted nodes – in the network. The different routing strategies and corresponding routing metrics to serve different application demands is also a valuable feature.

Table 2 - Properties of load balancing related work

Protocol	Type	Organization	Communication	Localization	Multi-sink support	Comment
DLBR	Routing	Distributed	Multi-hop	-	-	
GLBCA	Clustering	Centralized	Direct communication	√	√	
LCB	Clustering	Centralized	Direct communication	√	√	
NCLB	Routing	Centralized	Multi-hop	-	-	
OFFIS	Routing	Distributed	Multi-hop	-	-	Local flooding for dead-end resolving
LBSP	Routing	Distributed	Multi-hop	√	-	Narrow strip network topology
E3D	Routing	Distributed	Direct communication	√	-	
LEACH	Clustering	Distributed	Direct communication	-	√	
Arbutus	Routing	Distributed	Multi-hop	-	-	
RTLD	Routing	Distributed	Multi-hop	√	-	

Chapter 3

3. Partition-based Network Load Balanced Routing

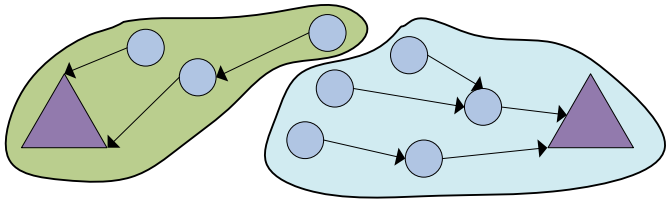
3.1 Partition-based Network Load Balanced Routing: A two-level approach

In the problem definition we have defined the demands of the new routing protocol. This chapter shows how these demands are analyzed and lead to the novel *Partition-based Network Load Balanced Routing* (P-NLB) protocol. The evaluation of the related works have shown that although many research haven been done on the topic of multi-sink WSN, they have resulted in only a few solutions for routing from single source to single sink in multi-sink networks. Also, none of the proposed load balancing routing algorithms has been designed for multi-sink networks, although this combination has great potential. P-NLB is a novel routing protocol for multi-sink WSN that uses a two-tier approach that combines metric-based routing on *local level* with a load balancing technique on a network wide *global level*.

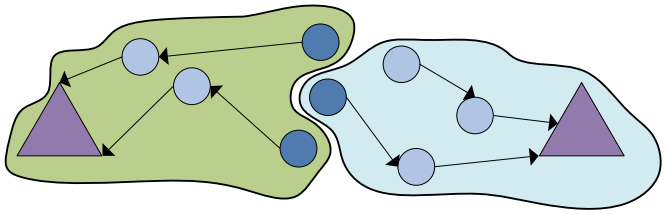
On a global level, a technique called clustering is used in order to spread the load in the network uniformly among all sinks in the network. The novel parts of this mechanism is that no explicit clustering phase is used, but *the nodes in the network achieve clustering on a global level, by clever routing on a local level*. Load balancing in a network is a NP-hard problem [32, 17]. On a global level the sinks determine the structure and cluster sizes of the network and provide the sensor nodes in the network with information about this network structure. To be more specific, if the clusters in the network are balanced and if not so, which cluster is the smallest. On a local level, the nodes use the information provided by the sinks in combination with local information to make their routing decisions. So the global level does not do the *actual* clustering or routing. It just gathers information from the network and provides the nodes with this information.

The local level does the actual routing and clustering and uses a metric based routing mechanism where every node decides for itself, what the next step is for creating a routing path and forwarding data over it. It does that by select a neighbour as the parent node, which forms a step in the spanning tree. The resulting spanning tree is used for routing the packets from nodes to sinks. The decision of selecting a neighbour as the parent node is not trivial. It depends on the information provided by the sinks on the global level – in case of the balancing mode – and the *routing strategy* and corresponding *routing metric*

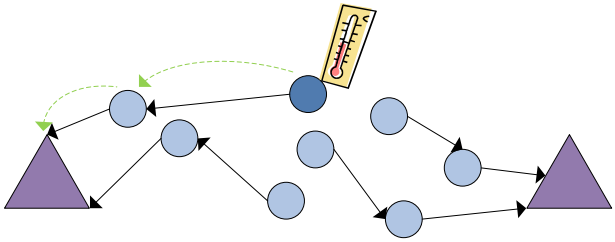
of the node. The routing strategy of a node – for example, avoid congestion – depends on the demands of the application running on the WSN. On the local level nodes use the cross-layered approach to exchange information with their one-hop neighbours and get a view of their *local* neighbourhood. An illustration of this two level routing approach is given in Figure 10.



a) On the global level, the network detects unbalance in the two cluster sizes



b) As a result, on the local level nodes adjust spanning trees, resulting in balanced clusters



c) Spanning tree is used for routing temperature reading from sensor node to data sink

Figure 10 – Two-level routing approach

In WSN, the most basic routing approach is the Shortest Path Routing (SPR) paradigm to send data packets to the sinks. SPR is defined as the routing mechanism where nodes forward data only to neighbours which are at a shorter distance –measured in hops – to the nearest sink. This results in a loop-free *Minimum Spanning Tree* (MST) rooted at a sink. However, in multi-sink networks, SPR does not guarantee that the resulting spanning tree is load balanced. It minimizes the number of hops a packet travels, leading to the formation of spanning trees containing different amount of sensors, since selecting the shortest path does not account for the effect of load aggregation on upstream links. Therefore, by assuming uniformly generated load per node, SPR creates spanning trees with different loads in the network.

Although the base point of P-NLB is SPR, it enhances this routing mechanism by uses routing metrics for more efficient spanning trees construction. P-NLB uses an approach which:

- Is distributed.
- Fully utilizes the existing of multiple sinks in the network.
- Does not need explicit network maintenance.
- Scales very well for large sensor networks.
- Needs no geographical location information.

Distributed

Each sensor decides for itself to which sink it will route its data; no centralized control of the sinks or other entity is needed for that. Sinks in the network will only have a small task of sending periodically information about the clusters into the network. This will be done implicitly using cross-layer communication and requires no broadcasts, keeping this mechanism scalable.

Multi-sink utilization

P-NLB features inter-cluster load balancing, by which the existence of the multiple sinks is used. Its intra-cluster metric-based routing leads to efficient routing and distribute the load within each cluster.

Network maintenance

There is no static routing spanning tree in the network; instead the routing tree is very flexible and adapts itself easily according to changes in the network. Therefore, there is no explicit maintenance necessary keeping the routing tree stable. We assume the target network is static – sensor nodes and sinks don't move through the network and no nodes are added or removed from the network. However, due to our flexibility we expect that P-NLB is also highly suitable for dynamic networks with mobile nodes and sinks. Mobility is not covered in this thesis, but part of the Future Work will investigate the performance of P-NLB in case of network mobility.

Scalability

Scalability is of great importance for P-NLB, because the target network type consists of many sensor nodes and data sinks. A solution that is not scalable would result in much communication overhead in the network and suffer from great unnecessary energy consumption. A distributed approach with low communication overhead is the only feasible approach to achieve scalability in the network. Therefore we will make extensively use of local information known by the nodes. We will get this information via cross-layer communication with the MAC layer, which is in our case the LMAC protocol. The LMAC protocol functions in such a way that nodes broadcast information about themselves to their one-hop neighbours. In this way they get a good view of their local network neighbourhood. No network wide broadcasting is used by P-NLB. Also, although do keep a neighbour table, they do not keep a routing table, only the parent node as forwarding vector is stored. They neighbour table does not increase when the network get larger, a routing table however, would increase in size.

Geographical location Information

While other algorithms assume the availability of GPS electronics on the sensor nodes or a localization algorithm, P-NLB needs no geographical location of the nodes. GPS location would make the sensor nodes both too expensive and energy demanding. A localization algorithm comes with a lot of communication overhead and is not always very reliable.

3.1.1 Adaptive Routing Mode with Cluster Size Distribution Detection

P-NLB defines two different routing methods to achieve an efficient routing protocol and enhance the basic SPR method. The first method takes as starting point the shortest path routing paradigm used in other current protocols. Instead of the randomized packet forwarding to any neighbour closer towards a sink, it uses well defined routing metrics to increase the efficiency of this method, similar to approaches in [6], [23] and [28]. In the rest of this document this method is called the *smart shortest path mode* (S-SPM). The other method is more complex and a novel solution. It is clustering based and has as goal balancing the traffic load over all the sinks in the network. This method is called *balancing mode* (LBM) in the rest of this document. Figure 11 displays these two routing modes within P-NLB. The difference between these two routing modes is the goal of these modes: the shortest path mode tries to achieve efficiency by using a very simple routing algorithm, while the balancing mode tries to achieve efficiency by looking at the clusters in the network and adjust these in order to balance the load in the network. Both modes intelligently use cross-layer information from the MAC layer, to achieve a higher efficiency, since less explicit communication between sensor nodes and sinks is needed. Which mode is used for routing in the network is determined by the setup phase, which is described in Chapter 3.2.1.

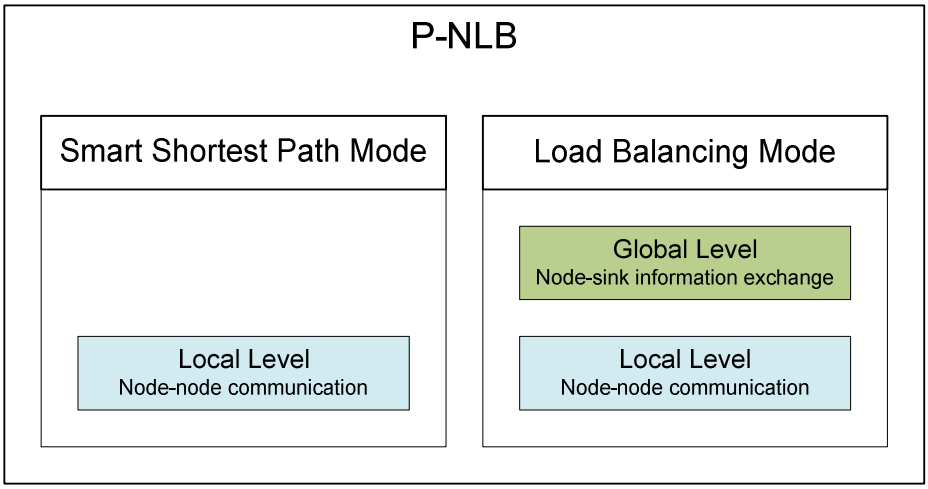


Figure 11 – Partition-based Network Load Balanced Routing Protocol

3.1.2 Summary

Now a short introduction to the features and routing mechanism of P-NLB is given and before continuing to a detailed analysis of the protocol a short summary of P-NLB:

- There are two routing modes, S-SPM and BLM, one of them is used in the network, depending on the initial cluster size.
- The global level provides the local level with clustering information.
- On the local level, nodes build spanning trees in network, each rooted at a sink and use this spanning tree to route data.
- Every node builds one edge of the spanning tree in an autonomously manner.
- It does this by selecting a parent, based on local information – in case of BLM also global information) and a routing metric.

The remainder of this chapter will explain in great detail how P-NLB works, starting with the protocol organization in Chapter 3.2, continuing with the global level in Chapter 3.3 and ending with the local level in Chapter 3.4. One example network is used for illustrating all features of P-NLB. This network

contains two clusters *A* and *B*. In some parts of this chapter, only a part of the network is used as illustration. The example network is shown in Figure 12.

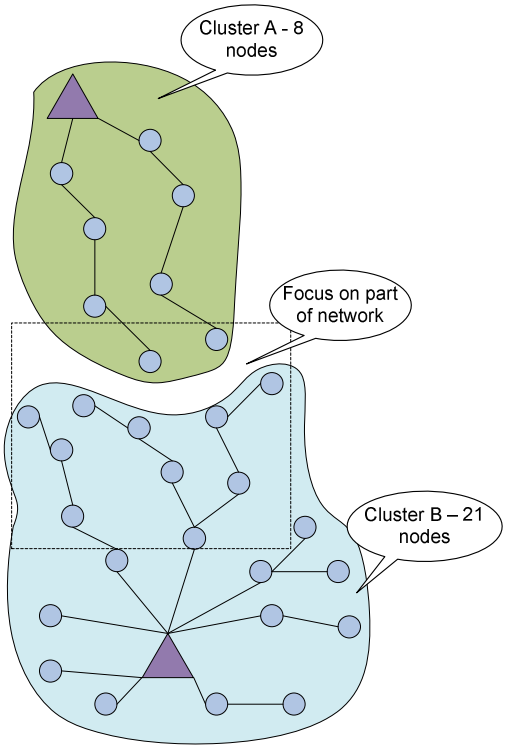


Figure 12 – Example network

3.2 Protocol organization

P-NLB consists of a setup phase and an operational phase.

3.2.1 Setup phase

In the setup phase, all the nodes in the network initialize with the LMAC protocol. They register to the network, learn about the neighbours in their proximity, get their hop count the sink(s) and finally acquire a LMAC timeslot. This phase is important for the nodes, because they acquire valuable information about their local network neighbourhood, which they use in the operational phase to immediately start efficient routing. After the setup phase has ended, the sinks in the network have information about the initial cluster sizes in the network, which is useful for determining the actual need of balancing the network. Section 3.3.3 gives more information about this.

The state diagram of the setup phase can be found in Figure 13. As can be seen in the figure, the network enters the operational phase in shortest path or balancing mode. This effects the operational phase of the nodes, which is explained in the next section.

The details of the LMAC initialization are not covered in this thesis. More details about the LMAC and LMAC initialization can be found in [7].

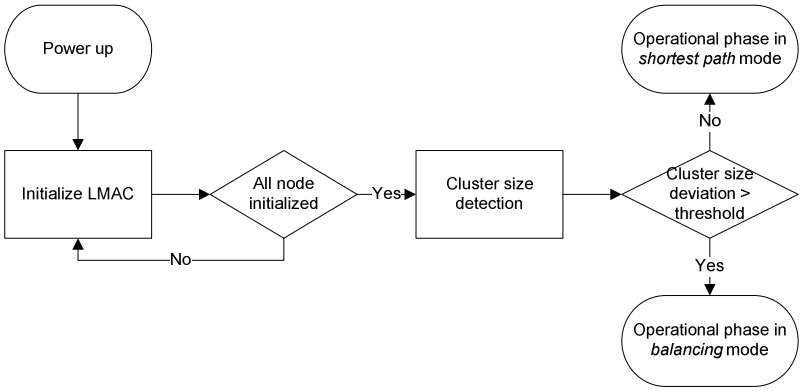


Figure 13 – State diagram of setup phase

3.2.2 Operational phase

In the operational phase, nodes already acquired the necessary information about their neighbours in the setup phase and now must establish a dynamic spanning tree to a sink, which can then be used for routing the data packets to the sinks. In this phase, the spanning tree to the sink is constantly maintained and adjusted to the most efficient routing paths in the network. This is done by the nodes as a result of constant updating of their parent nodes.

The state diagram of the nodes in the operational phase is shown in Figure 14. The steps belonging to the global level are coloured green, while the steps belonging to the local level are coloured blue. The state diagram clearly illustrates that the shortest path mode has only local level, while the balancing mode has also global level. The global and local levels of the protocol are further explained in the next chapters. The state diagram of Figure 14 doesn't specifically tell anything about receiving and sending data packets.

Details on updating the parent of the nodes can be found in Section 3.4 which describes the local level.

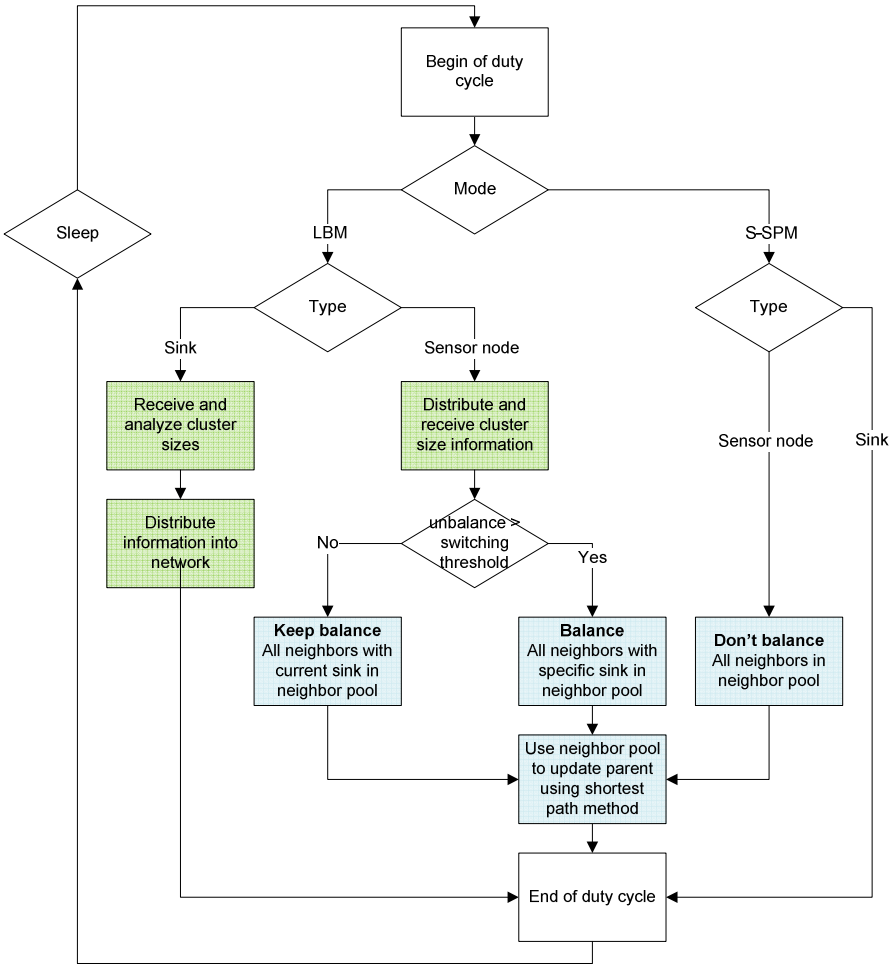


Figure 14 – State diagram of operational phase

3.3 Global Level – Cluster Information Gathering and Distribution

In Section 3.1 we have seen how the two-level approach of P-NLB combines local routing with global clustering. This paragraph takes a deeper look at the backgrounds of this clustering technique and the implementation in P-NLB.

3.3.1 Network clustering

On a global level this routing protocol will try to establish multiple non-overlapping spanning trees – also called clusters – in the network, each tree with a sink as root node. If all the spanning trees in the network contain more or less the same number of nodes, the network is balanced. If the generated traffic load in each sensor node is more or less equal, the traffic load in the whole network will be uniform. This objective of load balancing has previously been used in [16], [22], [30], [31], [32], [33], and [34], as already discussed in the Related Work chapter.

The first column of Figure 15 shows an unbalanced network, with two network clusters of 8 and 21 nodes in each cluster. In the third cell, the network has been balanced, with network clusters of 15 and 14 nodes in each cluster.

This goal of clustering is similar as that of LBC [22], but LBC uses an offline method for finding the ideal clusters size and composition, where the sinks gather information about the nodes in the network and calculate precisely what the best clusters are and which node is in which cluster. This approach is not very flexible, because in general nodes in WSN behave not very static and are prone to temporary and permanent failures. This leads to problems, because with each change the sinks must recalculate the best network partitioning and give this information to the sensor nodes. If the sinks are responsible for setting up the clusters they must know about any changes in the network and it takes time for them to adapt the clusters to these changes. Therefore P-NLB uses an approach where nodes themselves decide to which sink they route and therefore they know to which cluster they belong. They base these decisions mostly on local information supported by some global information from the sinks. Another novelty is that we will combine this clustering technique with the tree building routing technique, as used in for example ART. This combined algorithm is both flexible and scalable on global scale and provides efficient routing on a local scale.

The goal of the global level is providing the nodes in the network with information about the balance of clusters in the network.

3.3.2 Global network information gathering and distribution

In the balancing mode, the network will try to balance the nodes and traffic load uniformly among all sinks in the network which is much more complicated than the shortest path algorithm. In order to keep the network load uniformly distributed over the sinks, the sinks need to know what the actual network load is. They will give this information about the cluster sizes of every sink to the nodes in the network. The mechanism of collecting the information about cluster sizes and distribution to the nodes in the network has three steps. The algorithm is given in Algorithm I.

- **Information gathering.** Nodes keep track of the number of child nodes they have and aggregate and propagate this information to the sink at the root of it spanning tree. In this way, each sink knows what the amount of nodes in its spanning tree is and thus the cluster size.
- **Analyzing.** Assuming (direct) communication between sinks, each sink has information about all the other cluster sizes in the network and consequently the balance in the network
- **Distribution.** Distribute this information back into the network, using cross-layer communication.

Figure 15 illustrate these three steps.

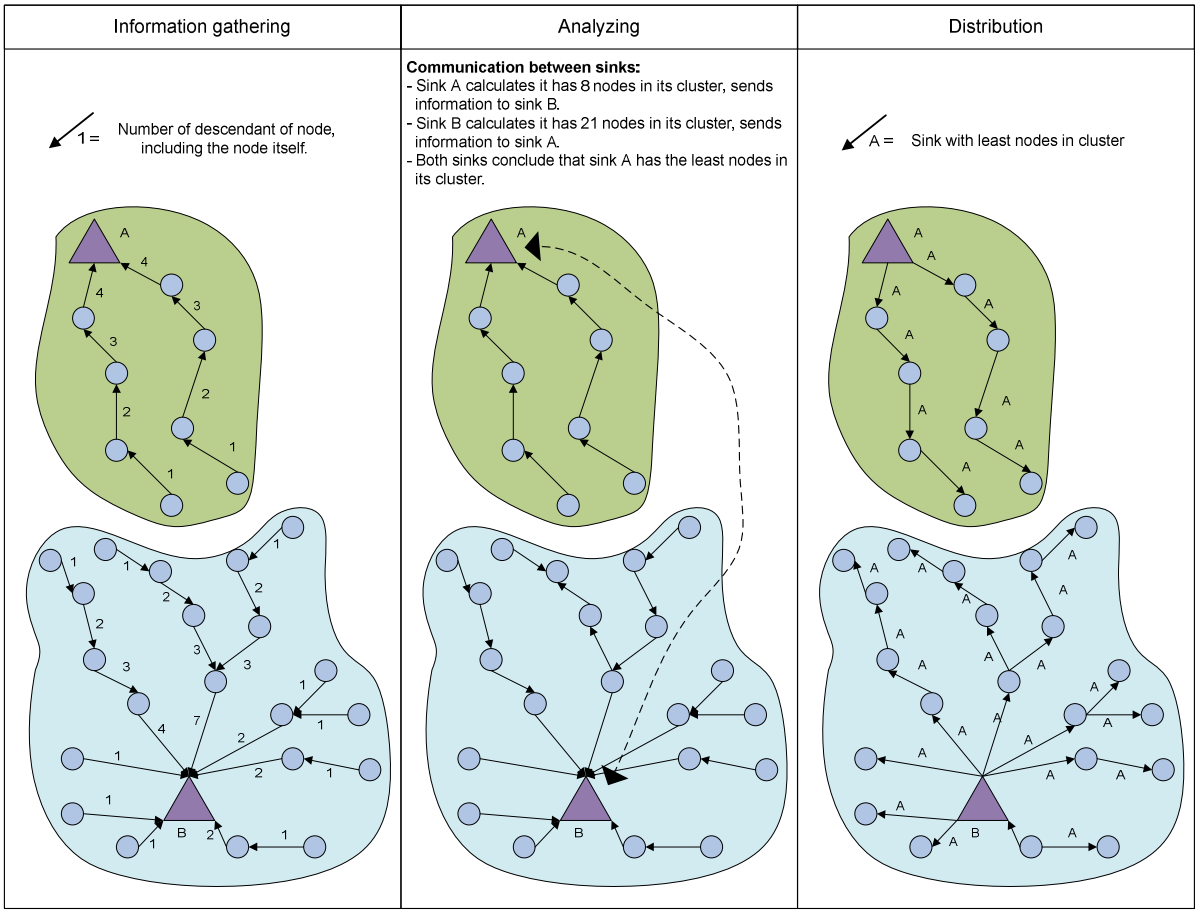


Figure 15 – Three steps of the global clustering algorithm

Algorithm I: Cluster information gathering and distribution

```
N ←set of all nodes;
Desc_i ← set of descendant nodes of node i
C_i ← set of child nodes of node i
P_i ← parent node of node i
S ← set of sinks in network
For each node i ⊆ N do
  //step 1: information gathering
  /* received descendant information of all child nodes, assuming node has any child nodes */
  RXDescChilds(C_i)
  Desc_i = 0
  for each c ⊆ C_i do
       Desc_i = Desc_i + Desc_c
    if i ≠ sink
       /* transmit updated descendant information to parent node, sinks have no parent nodes */
       TXDescParent(P_i, Desc_i)
  else
       //step 2: Analyzing
       /* send own and receive descendant information to / from other sinks
       for each s ⊆ S do
            TXDescSinks(Desc_i, s)
            RXDescSinks(Desc_s, s)
       /* calculate with information of all sinks which cluster is the smallest */
       SC ← CalcSmallestCluster(Desc_s, Desc_i)
  //step 3: Distribution
  for each c ⊆C_i do
       /* send SC to child node, assuming node has any child nodes */
       TXSCChild(c, SC)
 End
```

It is a continuous process of gathering, analyzing and distributing the information, there are no specific phases. The drawback of this continuous process and using cross-layer information is that it takes time for the information of the nodes to reach the sinks and it also takes time for this information to reach all the nodes in the network. As a result this global information is not always up-to-date and not always very reliable. Due to changes in the network – nodes update and continuously parent switching – it is hard for the sinks to get a *complete* view of the current network topology. This is however, not a major problem and a common phenomenon in distributed algorithms. Several factors influence the reliability of this global information.

- The average hop count of the nodes in the network: If the hop count increases it will take a longer time for the information to reach the sink, and during this time, the chance increases that the information is no longer valid.
- The rate at which nodes in the network update their parents.
- Other factors, such as node mobility and temporary and permanent node failures.

As a results of this incomplete network view and not up-to-date information nodes might take not the best decisions and sub-optimal spanning trees are build in the network. So there is a trade-off between de-centralization and optimal routing paths, which might have a negative effect on the performance of the protocol.

3.3.3 Initial network topology cluster detection

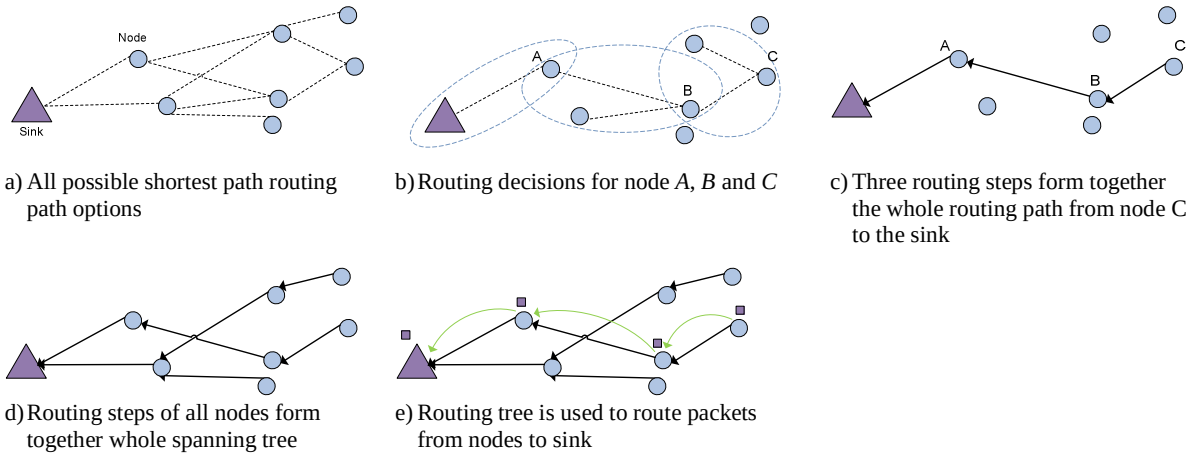
The balancing mode of P-NLB performs best in networks which have the typical asymmetric shapes with large and smaller clusters. In case of the shortest path mode, many nodes in the larger clusters route to a few sinks, while only a small part of the nodes routes to the other sinks, which causes congestion in the large clusters. On the other hand, in balanced network types, shortest path mode outperforms the balancing mode. Therefore, it is important to use the right routing mode in the right network type, and detecting the network shape is essential.

A method for detecting the size of the clusters of nodes around the sinks is using the hop count information of the nodes. During a detection phase, all nodes send a short message to the nearest sink informing it about its presence. In case of multiple nearest sinks it will send a message to only one of the sinks, so that all nodes are known by only one sink. All the nodes that are close to a specific sink and report to this sink belong to the network cluster of that sink. This is not the same cluster as the routing spanning tree cluster, described earlier, since there is not yet a routing tree. Next, all the sinks exchange information about the sizes of their network clusters and some measure to detect the dispersion – the standard deviation for example – is calculated by the sinks. A high standard deviation indicates that the nodes in the network are not uniformly distributed over the sinks in the network.

3.4 Local level – Optimized Metric-based Routing Tree Building

We have seen the mechanism where on a global level sinks provide nodes with the right information about the cluster size so the nodes can on a local level influence the cluster sizes while updating their parents. We have also seen that only LBM has this global level while S-SPM does not. This section explains how the routing tree building and data forwarding mechanism on local level works and what nodes do with the information provided by the global level.

The local level does the actual routing and uses a metric-based routing mechanism where every node decides for itself, what the next step is for creating a routing path and forwarding data over it. The selected neighbour is called the *parent node* and the forwarding node itself is called the *child node*. All these small one-hop routing paths will eventually result in one long routing path from source node to the sink. All the routing paths from all source nodes to a specific sink form the *spanning tree*, also called *routing tree*, for that sink. Every sink is a part of a unique non-overlapping spanning tree. This spanning tree is highly dynamic, because nodes continuously update their parent according to changing conditions in the network. The dynamic routing tree is used to route the packets from sensor nodes to sinks. Figure 16 illustrates in five steps how a spanning tree is constructed in the network and used for routing packets.



a) All possible shortest path routing path options

b) Routing decisions for node *A*, *B* and *C*

c) Three routing steps form together the whole routing path from node C to the sink

d) Routing steps of all nodes form together whole spanning tree

e) Routing tree is used to route packets from nodes to sink

Figure 16 – Building routing tree from small steps

The decision of selecting a neighbour as the parent node is not trivial. It depends on the information provided by the sinks on the global level – in case of the balancing mode – and the *routing strategy* and corresponding *routing metric* of the node. The routing strategy of a node depends on the demands of the application running on the WSN. On the local level sensors use the cross-layered approach to exchange information with their one-hop neighbours and get a view of their *local* neighbourhood. They will use this information to analyze their neighbours and based on the routing metric select the best neighbour as its new parent. The information the nodes have about the neighbours in their local neighbourhood is the following, where the last four data is needed for the four routing metrics:

- Node ID
- MAC hop count
- Routing hop count
- Sink which forms the root of the spanning tree the node is in – in order words, the cluster
- Number of child nodes
- Estimation of the number of descendant nodes
- Buffer occupancy
- Energy level

3.4.1 Demands, routing strategies and routing metrics

The application running on the WSN has *demands* on the WSN. A long lifetime, low message latency or high throughput for example. Based on these demands different *routing strategies* might be used, each strategy best suited for a specific demand. A routing strategy can be: avoid low-energy nodes, avoid congested nodes, route to closest sink. Different routing strategies might used separately in a single

network, if there are high- and low-priority messages for example. Based on the routing strategy, a node can choose a *routing metric* to select a parent, and form a spanning tree in the network.

It might be possible to have different demands in the same network. Nodes in such a network generate with a certain rate messages with status information about their environment, where there focus lies on an energy efficient network, with a long network lifetime. On the other hand, some nodes might periodically have some high priority messages, which must make it to a sink as fast as possible, so latency is an important factor for these messages. Disparate priority messages are not covered in this thesis, but part of the Future Work.

Simulations must clarify which routing metrics fits which demand best. Four routing metrics are used in this thesis, although one could define more, i.e. link quality, link usage, neighbour distance.

- **Child nodes**. The degree of the routing spanning tree in the network is of important value. If a node has many child nodes, it receives (small) data packets from each child node within one frame. This has several negative consequences for this node. First, all this data needs to be stored in a buffer before it can forward this to its parent. Remember, every node can occupy only one timeslot for forwarding data, while it is likely to have more than one child nodes that transmits data to it. Receiving more data than it is able to transmit results in full buffers and eventually buffer overflows. Second, every time it receives data its transceiver consumes energy, and when it forwards the data its transceiver is again activated. If a node has many child nodes, it is likely that it consumes more energy and its energy source gets depleted earlier. For those reasons, looking at the number of child nodes a neighbour already has it a good routing metric.
- **Descendants**. The number of descendants a node has can also be taken as a routing metric. This metric is related to the previous metric *child nodes*, but differs slightly. Where the metric *child nodes* takes only into account which neighbours are directly after the node on the routing path, descendant nodes are *all* the *upstream* nodes on the routing path. This mainly affects the amount of traffic a node has to process. A node with many descendants, all generating sensor data packets, has to receive and forward all that data.
- **Energy level**. After a certain uptime of the network, some nodes might have to forward/transmit a lot of data; this will drain their energy source. Other nodes might stay in an idle state for a long time, or transmit only hardly any data. Some nodes might even have an unlimited power supply. It is wise to transfer traffic from the nodes with a near empty energy buffer to the nodes with a full buffer in order to extend the total network lifetime.
- **Buffer**. Every node has a message buffer where it stores the incoming packets, before they are forwarded to the parent of the node. Assuming all packets have the same priority, and a first in first out strategy is used, the latency of packets increases when there are many packets in this message buffer. Favouring nodes with empty buffers over nodes with full buffers is a good routing metric.

3.4.2 Parent Selection Mechanism

Section 3.4 explained how a routing tree is build and used for packet forwarding. Now the *parent selection mechanism*, the mechanism where each node builds one step of the routing tree, is described.

Since we assume the network is a connected graph, all nodes have at least one neighbour. However, in most cases every node has several neighbours, which all can be selected as parent node for forwarding data to. Selecting the parent node is not trivial; it has a significant effect on the resulting path from source node to sink. The most basic approach for selecting a parent is the shortest path mode; any neighbour which is closest to any sink is selected. In this case the node will consider any neighbour as a candidate and pick the one closest to any sink. This shortest path mode results in a shortest path spanning tree from

nodes to a sink, every sink is the root node of another spanning tree. However, basic SPR is not very efficient and the load balancing mechanism of LBM abandons the shortest path paradigm. Therefore a metric-based parent selection mechanism based on SPR is introduced which adds an extra step, neighbour pool construction, to the beginning of the parent selection. This neighbour pool construction is essential for LBM, because it implements the load balancing technique in P-NLB. The difference between SPR and P-NLB is that the latter adds a metric-based parent selection mechanism, which results in efficient selection of the shortest path parent, instead of a randomized selection. Also, in S-SPM *absolute* shortest path routing is used – always route towards the closest sink. On the other hand, in LBM shortest path routing *within the cluster* is used; within the current cluster of a node the parent node is always closer towards the cluster head – the sink – although a sink of another cluster might be the closest sink. In Figure 19 e) a network is balanced using LBM and within the green cluster shortest routing is used, although for some nodes in the green cluster the sink in the blue cluster is closer by.

The parent selection mechanism has four steps.

- One step for defining neighbour pool with use of global information (in case of LBM) and local information.
- Three steps for applying routing metric to neighbours in neighbour pool and select neighbour as new parent.

The algorithm of the parent selection mechanism is given in Algorithm II. In the next section of this chapter, all these four steps of the parent selection mechanism are explained in detail. Also three examples are given: one of neighbour pool construction and two of the parent selection mechanism in both S-SPM and LBM.

Algorithm II: Neighbour pool construction and parent selection

```
N ←set of all nodes;
P_i ← parent node of node i
NB_i ← set of neighbours of node i
NBP_i ← neighbour pool of node i
node i ⊆N
U ← Cluster size unbalance
ST ← switching threshold
Cli ← cluster node i belongs to
CS ← smallest cluster
//step 1: neighbour pool construction
if routing mode == shortest path
   /* all neighbours are in neighbour pool
    NBP_i = NB_i
else
     /*routing mode is balancing, check switching threshold*/
     for each nb ⊆NB_i do
          if ST > U
               /* balance clusters*/
               if CL_i == CS
                  AddNbrToNBP(NBP_i, nb)
          else
             /*stay in same cluster, don't change clusters */
             if CL_i == CL_nb
                  AddNbrToNBP(NBP_i, nb)
End else
//step 2: Check hop count, discard neighbours which have not the lowest hop count */
NBP_i  ← CheckHC(NBP_i)
//Step 3: Apply metric on neighbour pool
NBP_i  ← ApplyMetric(NBP_i)
//Step 4: Parent selection
P_i ← SelectParent(NBP_i)
 End
```

3.4.3 Using global and local information to define neighbour pool

In LBM, the nodes have received information from the sinks in the network about cluster sizes, and in particular, which cluster is the smallest. Since the load in a cluster is equal to the number of nodes in the cluster, a node knows which cluster has the lowest load. A node also knows in which cluster it is located – this information is passed from the data link to the network layer – and in which clusters its neighbours are located. A node can choose to leave its current routing tree (cluster) and join another cluster by selecting a neighbour as its new parent, if is in another cluster. Of course, this is only possible if at least one of its neighbours is located in another cluster. By joining another cluster nodes can decrease the size of their own cluster and increase the size of neighbouring clusters, and thus balance the load in the clusters – *inter-cluster load balancing*. The neighbour pool construction mechanism filters all neighbours of a node for a specific cluster, before the metric-based parent selection takes place. There are three categories of neighbours:

- Neighbours that are in the same cluster as the node
- Neighbours that are in the cluster which has the smallest cluster size
- Neighbours that have neither of both conditions

Constructing the neighbour pool is done in one step in S-SPM and two steps in LBM:

- **Step 1a.** Both modes: get all one-hop neighbours.
- **Step 1b.** In LBM only: if there are any neighbours located in the smallest cluster and the *switching threshold* is not met, remove all neighbours which are not in that smallest cluster.

Switching threshold

When, in a two cluster network, two clusters A and B have almost the same sizes, for example 5 and 6 nodes respectively, nodes at the border of these two clusters will still try to balance these clusters, although this is not possible due to the uneven number of total nodes in the two clusters. A certain Node *n* located in Cluster B but close to Cluster A will try to join Cluster A, because it is the cluster with the smallest amount of nodes. As a result Cluster A has now 6 nodes, one more than Cluster B. Now this Node *n* notices that cluster B has the smallest amount of nodes and will try to *Cluster A* again. In this small example network this *oscillation* has not much effect, but in larger networks which more nodes in the middle between two clusters it will cause instability and result in decreased performance. In order to counter this oscillation the parameter cluster size threshold is introduced, which must stop nodes from attempting to balance slightly unbalanced networks. This parameter is important for determining the correct neighbour pool. If a node receives information from the sink that a cluster is unbalanced it will not attempt to balance this cluster if the unbalance is smaller than *switching threshold*. In Table 3 is a list of parameters for determining the correct neighbour pool is given.

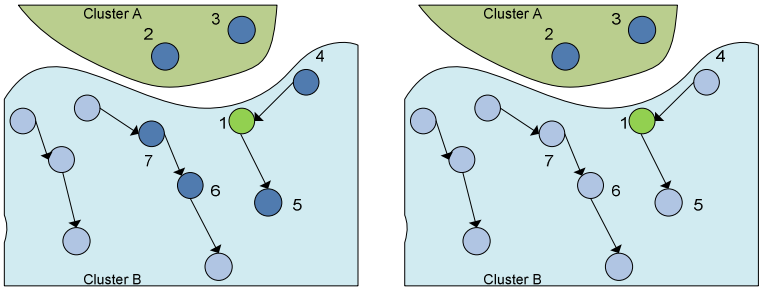
Table 3 – Requisites leading to right neighbour pool

Mode	Switching threshold < cluster size	Switching threshold < cluster size
S-SPM	Neighbours in any cluster	Neighbours in any cluster
BM	Neighbours in smallest cluster	Neighbours in same cluster

Example of neighbour pool construction

After network initialization there are two spanning trees, *Cluster A* and *Cluster B*, in the network as shown in Figure 12. A close-up of a part of the network is shown in Figure 17. In this example, which illustrates Table 3, *Node 1* – green in the figure – updates its parent. *Node 1* has six neighbours – *Node 2*, *Node 3*, *Node 4*, *Node 5* and *Node 6* – which are dark blue coloured. In Table 4, the consequences of these parameters for each neighbour of *Node 1* are given. If the network is in S-SPM mode, cluster sizes do not play a role and all neighbours of Node 1 are in the neighbour pool, as shown in Figure 17 a). In LBM mode this is all different since the cluster to which each neighbour belongs is important. For example

Node 2 belongs to C*luster A* and *Node 1* has received the information that *Cluster A* is the smallest cluster. This means that *Node 2* belongs to the correct neighbour set of *Node 1*, because the cluster deviation is larger than the threshold. *Node 3* is the only other node that together with *Node 2* belongs to the correct neighbour pool of *Node 1*. The neighbour pool of *Node 1* in LBM is shown in Figure 17 b).



a) Step 1a. Get all one-hop neighbours

b) Step 1b. LBM only. Remove neighbours which are not in smallest cluster *A*

Figure 17 – Example of neighbour pool construction

Table 4 – Example of neighbour pool construction

Neighbour	Cluster	S-SPM		LBM	
		Switching threshold < cluster size	Switching threshold < cluster size	Switching threshold < cluster size	Switching threshold < cluster size
2	A	√	√	√	-
3	A	√	√	√	-
4	B	√	√	-	√
5	B	√	√	-	√
6	B	√	√	-	√
7	B	√	√	-	√
√	means: place neighbour in neighbour pool		-	means: do not place neighbour in neighbour pool	

3.4.4 Using local information and neighbour pool to select a parent

Now the nodes have received information from the sinks about cluster sizes and used this information together with local information to determine their correct neighbour pools. In the next phase a parent is selected from the neighbour pool and the routing tree is build/adjusted. Since the neighbour pool construction regulates the load balancing of LBM, the metric-based parent selection is very simple and equal for both S-SPM and LBM. Selecting a parent consists of three steps:

- **Step 2.** Check hop count of neighbours; only consider neighbours with the lowest hop count in next steps.
- **Step 3.** Apply routing metric on the remaining neighbours. If routing metric is *Child Nodes*, it only keeps the neighbours with the smallest amount of child nodes. If routing metric is *Buffer*, it keeps only the neighbours with the least amount of packets in their buffers, etc.
- **Step 4.** All neighbours left have the same properties and one random neighbour is selected as the parent.

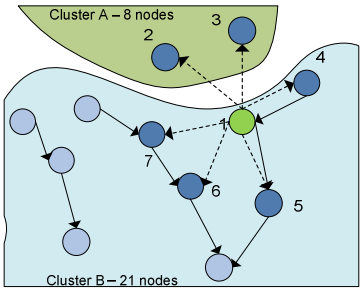
Two examples of this parent selection process will be given: one for S-SPM and one for LBM. In both examples *Node 1* updates its parent and the local information it has about its neighbours is shown in Table 5.

Table 5 – Balancing parent select example: neighbour properties

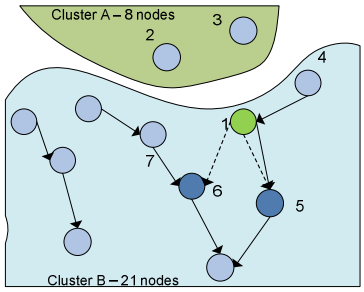
Neighbour	Cluster	Hop Count	Routing Metrics			
			Child Node	Buffer	Energy Level (%)	Descendants
2	A	4	0	3	80	0
3	A	4	0	1	71	0
4	B	4	0	8	56	0
5	B	2	0	5	88	2
6	B	2	1	3	92	2
7	B	3	1	1	57	1

S-SPM parent selection example

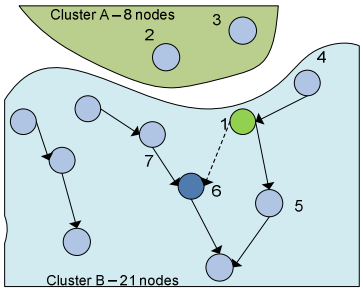
In this example none of the nodes have updated their parents yet, so the network is still unbalanced with *Cluster A* containing 8 nodes and *Cluster B* containing 21 nodes, but because the network is in S-SPM it doesn't matter that the network is unbalanced. Figure 18 shows a part of the network with *Node 1*, which is going to update its parent, and its six neighbours in it. The nodes have the routing metric *Buffer*. *Node 1* has six neighbours in its neighbour pool – *Node 2*, *Node 3*, *Node 4*, *Node 5* and *Node 6* – and the information it has about those six nodes is listed in Table 5. *Node 5* is the old the parent of *Node 1*. Figure 18 explains why *Node 6* is selected as the new parent, from the neighbour pool of six nodes. The new parent of *Node 1* is in the same cluster as the old parent of *Node 1*, therefore, the clusters do not change.



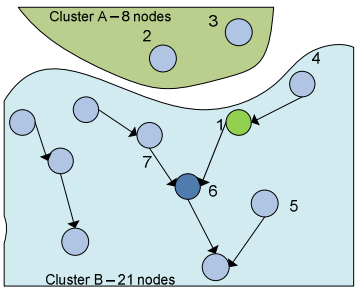
a) Neighbour pool – blue nodes – has been constructed in Step 1



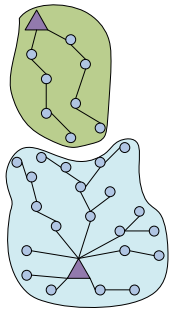
b) Step 2. Nodes 5 and 6 have lowest hop counts; other neighbours are dropped as candidates



c) Step 3. Node 6 has smallest buffer occupancy; node 5 is dropped as candidate



d) Step 4. Node 6 is only node with smallest buffer occupancy, therefore it is selected as the new parent



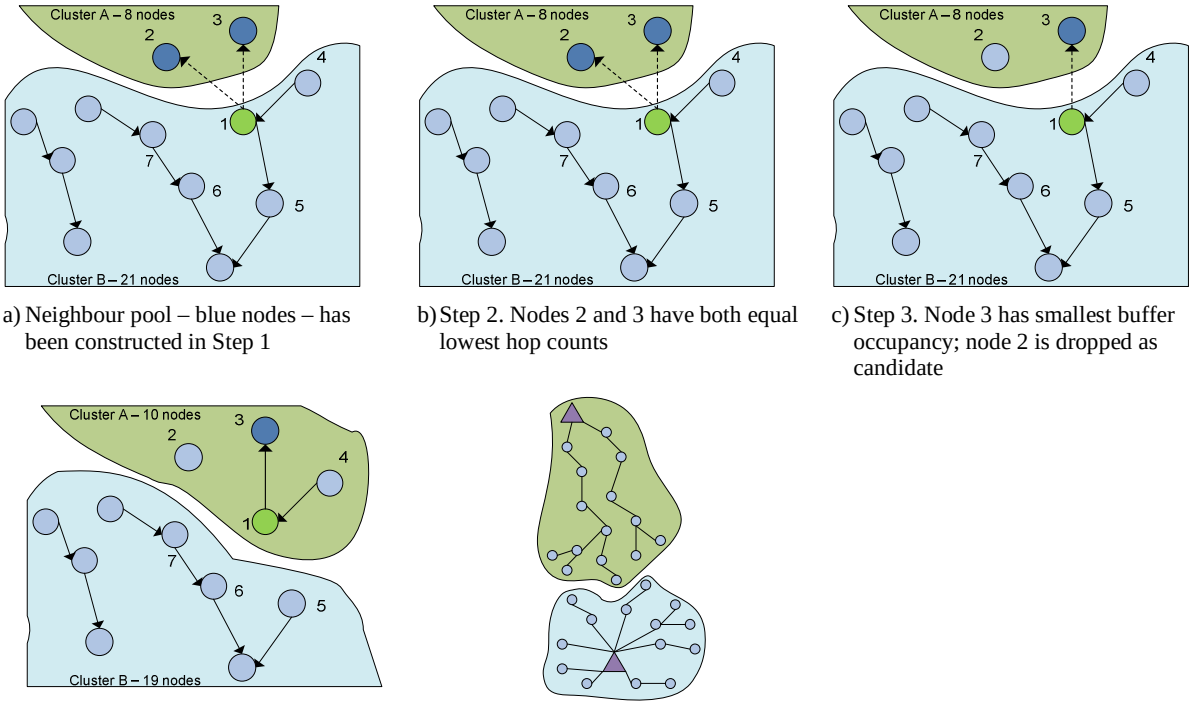
All nodes update parents. Cluster is still unbalanced – 8 and 21 nodes.

Figure 18 – Shortest path example

LBM parent selection example

In this example, none of the nodes have updated their parents yet, so the network is still unbalanced with *Cluster A* containing 8 nodes and *Cluster B* containing 21 nodes. Figure 19 shows a part of the

network with *Node 1* and its six neighbours in it. The nodes have the routing metric *Buffer*. *Node 1* has two neighbours in its neighbour pool – *Node 2* and *Node 3* – and the information it has about those two nodes is listed in Table 5. The neighbour pool construction step already filtered out the other neighbours, since those neighbours are in *Cluster B*, which is not the smallest cluster. The parent selection mechanism is equal to that in the S-SPM example, but the neighbour pool is different. As a result the new parent, *Node 3*, is also different.



a) Neighbour pool – blue nodes – has been constructed in Step 1

b) Step 2. Nodes 2 and 3 have both equal lowest hop counts

c) Step 3. Node 3 has smallest buffer occupancy; node 2 is dropped as candidate

d) Step 4. Node 6 is only node with smallest buffer occupancy, therefore it is selected as the new parent. Cluster sizes changes a bit.

e) All nodes update parents. Cluster sizes are now as equal as possible – 14 and 15 nodes – in other words, the load is balanced.

Figure 19 – Balancing mode example

3.4.5 Routing mechanism optimizations and other issues

Node and area failures

Shortest path routing (SPR) is the most basic routing method and leads to the construction of routing trees which are of the form *minimum spanning tree* (MST) – assuming the link between every pair of nodes within transmission range is of equal weight. However, SPR is not the most flexible method when looking at a common problem in WSN – node and area failures – and techniques which can be used to overcome this problem. Nodes in WSN are relatively inexpensive devices which must function in a harsh environment, while having only a limited power supply. Link quality is often very variable, which makes communication in a WSN very prone to errors, which can be temporary due to environmental conditions or permanent due to hardware level failure of the inexpensive nodes or power source depletion. Another form of node failures are the area failures in the network, where communication in a whole group of node can be disrupted, due to environmental conditions. A heavily congested node in the network could also be considered as a – temporarily – failed node, and should also be bypassed if possible. Without proper

precautions, these node and area failures can lead to unacceptable congestion in the network. A routing algorithm which doesn't adapt to the failure in the sending path of the packets will suffer from congestion and packet loss in the node in front of the failing node. An efficient routing algorithm however must be able to detect such a failure and find a path around this failing node.

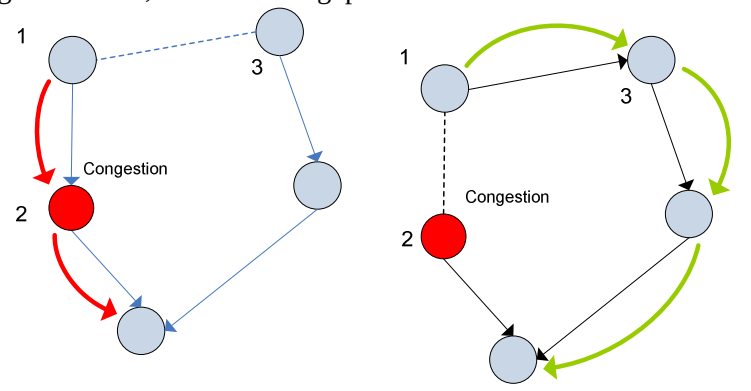
Link- and node failures are often caused by the unreliable wireless medium and (temporary) node failures, rather than congestion, which is most times the case in traditional wired networks. [25] [26] and [28] all describe this phenomena and methods for dealing with this problem. There are basically three methods for overcoming node failures:

- Periodically maintaining and, if necessary, rebuilding the routing path. In most cases, completely rebuilding a broken path costs a lot of communication and thus energy.
- Using multiple paths from source to sink to route data to. In case of node failure on one path, the same packets on the other path(s) are not affected by this and safely arrive at the destination.
- Probabilistic selecting a routing path for forwarding packets. If a node fails on a path, not all packets are blocked, since a next packet will, with a certain probability, take another path.

Although the considered network in this thesis is static without node failures, the temporary "node failures" caused by congestion are likely to occur. Looking at the four defined routing metrics, the routing metric *Buffer* is partially able to bypass such congested nodes. If it can choose between two equal neighbours, one with an empty buffer and one with a full buffer, it can choose the node with the empty buffer. However, sometimes this is not possible if there if the only neighbour has a full buffer. Therefore some other measure for dealing with congestion is very useful. The mechanism P-NLB uses is allowing nodes to "loosen up" the shortest path paradigm a little.

Shortest Path Routing relaxation

Since P-NLB has already a highly flexible routing tree, another method is used for avoiding congested nodes. The shortest path paradigm is relaxed a little bit to achieve better results. Instead of always selecting a neighbour closer to a sink, a node might select a neighbour which has the same hop count as the node itself has. With this small relaxation of the shortest path constraint, bottlenecks can be better avoided in the network. The cost of this is only small, a slightly longer routing path, and the need for some small precautions in order to avoid loops in the network. A small example of this shortest path routing relaxation is given in Figure 20. Simulation results in Appendix I Figure 35 show the protocol performance with and without this shortest path relaxation. As shown in that figure, the latency increases a bit, since path length increases, but also throughput and network lifetime benefit from this mechanism.

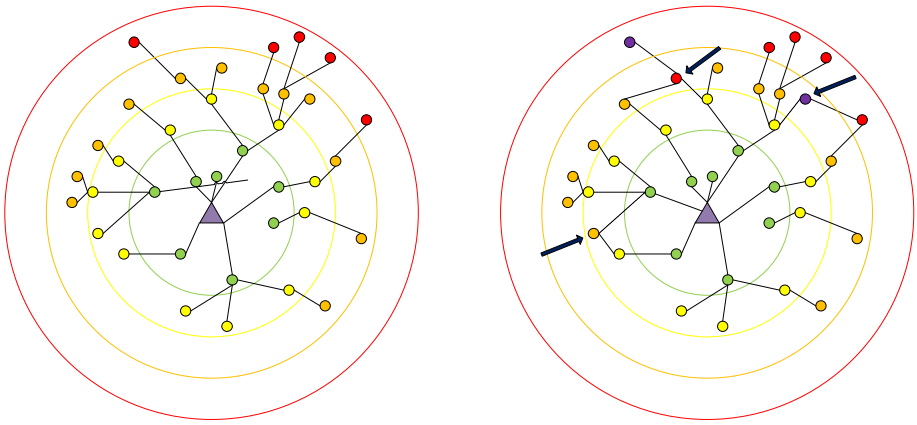


a) Node 1 notices congestion in parent Node 2 on routing path

b) Node 1 routes around congestion by selecting parent Node 3 with equal hop count; path length increases

Figure 20 – Shortest path routing relaxation

In LBM nodes do not follow pure shortest path routing, since nodes does not always build the spanning tree towards the closest sink, but also take cluster in consideration. Therefore a node has not only information about its distance – measured in hops – to the closest sink, the *MAC hop count* (MHC), but also the level of the node in the routing tree, the *routing hop count* (RHC). The RHC is different from the MHC if the shortest path mechanism is not used. An example of the differences between MHC and RHC is given in Figure 21 where in Figure 21 a) the four RHC levels of the nodes are shown in four different colours. The RHC is equal to the MHC. In Figure 21 b) three nodes don't use the shortest path routing method anymore and select parents that have an equal or higher MHC than they self have. As a consequence the RHC of those two nodes and their child nodes is different than their MHC. They nodes with a different RHC are marked by the arrows in Figure 21 b). In LBM nodes deliberately route away from the closest sink, towards another sink if this is necessary for maintaining the load balance in the network. By doing this, the difference between MHC and RHC increases. Simulations show (Appendix I, Figure 32) that initially nodes benefit from this increase, but when this difference increases too much, it turns into a decreased performance.



a) MAC hop count levels b) Routing hop count differs from MAC hop count

Figure 21 – Two different hop count definitions

Loop detection and avoidance

When using LBM, loops can be created in the network. This must be prevented as much as possible, and if such a thing nevertheless occurs, the loop must be broken and the correct routing path must be restored. There are several methods for detecting and preventing loops in the network and each of these methods has its own characteristics.

- **Sequence numbers of packets.** Packets keep track of the number of hops they travelled through the network, and store this sequence number in the packet. If a node notices that the sequence number of a packet has risen above a certain predefined threshold, it assumes there is a loop in the network. An example of a protocol where this method is used is Ad-hoc On Demand Distance Vector routing (AODV) [35]. The drawback of this method is that it requires additional space in packets for storing the sequence number.
- **Tracking hop counts.** A various on the previously described method, is keeping track of the hop count of the node's location in the routing path, instead of the sequence number of the packets. If nodes detect a (constant) increase in hop count, or the hop count rises above a certain threshold, they assume their routing path contains a loop and they break this loop and repair their routing path. This is slightly more efficient because the hop count information is not stored in the packets, which is relatively expensive, but it is stored in the nodes, which have more resources. Although this method detects loops, it doesn't prevent them from being created in the first place. In [1],

they use a similar approach, where a node snoops forwarded packets in checks if it detects it is the originating source node.

- **Shortest path spanning tree.** Nodes form a shortest path spanning tree in the network, with the sink as the root node. Because nodes always select the shortest path to the sink, a loop cannot be created because than it cannot be the shortest path. Assumes a static network where nodes don't change position. Easiest method of all, guarantees loop-free routing, but limits the routing possibilities and efficiency of the routing protocol.
- **Connected paths.** Nodes detect and select only parent nodes which are connected to a sink. Loops cannot be formed in the network, because nodes would not connect to such a loop, because it would not be connected to a sink. Nodes need up-to-date information about their status, but propagation of this information through the network might take a long time; therefore, this method cannot adapt very fast to changes in the network topology. This method allows more various in the routing paths of nodes and more possibilities of avoiding bottlenecks in the network.
- **Assigning credits.** Nodes form a routing path towards the sink, by selecting parent nodes which are closer to the sink, but the routing path are given credits, which can be used to deviate from the ideal shortest path to sink. It does not completely prevent loops, but avoids them. This method is often used in multi-path routing protocols. It needs an additional method for detecting loops if these are created nevertheless.

In S-SPM, loops are not an issue, because due to the shortest path paradigm loops cannot be created in the network. On the other hand, in LBM, precautions are needed to detect and avoid loops in the network. In P-NLB loops are caused due to outdated local information about neighbours. Nodes change their parent constantly and therefore, the routing paths in the network also change. However, it takes some time for this information to reach all the nodes on the routing path and the neighbours of these nodes. P-NLB uses the technique of tracking routing hop counts to detect loops in the network. If a nodes detects a loop in the routing path, the path is broken and a new (loop free) path is established. In that case, it will set a back-off timer and while this timer counts down to zero, the node is able to receive updated information about its local neighbourhood. When the back-off timer reaches zero, it will again select a new parent.

Parent update rate

The rate at which nodes update their parent has an influence on the performance of some routing metrics of P-NLB.

- Updating the parent involves some computations, leading to (minor) energy consumption.
- Influences the stability of the routing trees.
- Affect "lifetime" of some of the local information.

If a node updates its parent at a fast rate, for example every round, the routing tree structure also changes fast. For example, the information a nodes has about the amount of descendants of its neighbours is outdated if the routing path changed in such a way that those child nodes are no longer descendants of that neighbour. Consequently, low parent update rates increase the lifetime of local information. On the other hand, the occupancy packet buffer of a node changes very fast and a node might select a neighbour with lowest buffer occupancy; however, two rounds later this buffer occupancy might be much higher. If it does not update its parent, it will have the neighbour with the highest instead of the lowest buffer occupancy.

In LBM, there is another important issue related to the parent update rate. The information gathering, analysis and distribution mechanism on the global level is even more dependent on the correct information of the network. A sink determines its cluster size by looking at the number of descendants of its neighbours, which is basically the size of the routing tree. If the information it receives by its

neighbours is very different from the actual situation, it estimates the wrong cluster sizes and the load balancing mechanism does not work correctly. As mentioned earlier in Section 3.3.2, this information is never completely up-to-date. Simulations with varying parent selection rates confirm this. Results of these simulations can be found in Appendix I, Figure 33 and Figure 34.

Chapter 4

4. Evaluation

Simulations are needed to test the performance of the new protocol, in comparison with existing protocols. Direct implementation without simulating first is very time consuming. Setting up a test-bed of nodes costs a lot of time, which is wasted if there are still design flaws in the protocol. Simulations of this work are performed using MATLAB as programming tool. MATLAB is used instead of common network simulators such as Omnet++ and NS2, because it is more suited for simulating large networks with many nodes. Also, MATLAB has better visualization tools than other high-level programming languages such as C++ and JAVA. Both routing modes of P-NLB, S-SPM and LBM, are simulated. In the S-SPM routing mode simulations, the network is always routing using S-SPM – without load balancing. In the LBM routing mode, nodes use the cluster size distribution detection mechanism in the setup phase to enter the operational phase in S-SPM – without load balancing – or LBM – with load balancing. Besides those two routing modes of P-NLB, SPR and the Node Centric Load Balancing (NCLB) protocol are also implemented in our simulator and used in the simulations. SPR acts as a lower bound reference of what the performance of a basic not optimized routing algorithm would be. NCLB is a centralized algorithm and adding it to the simulations allows a comparison of a distributed with a centralized algorithms. However, as mentioned before, the load balancing problem is a NP-hard problem, and NCLB provides no hard upper bound, but only an approximation. Goal of the simulations is verification if the targets of the novel routing protocol are met. Also, the best routing metrics for the application demands *high network lifetime*, *low latency* and *high throughput* must be found.

4.1 Network and simulation setup

Network & simulation parameters:

- **Number of sensor nodes and data sinks.** As simulation duration increases exponentially with the number of nodes, there is a limit of 64 on the number of nodes in the network. The number of sinks in each network depends on the number of nodes in the network, as each sink can handle only a certain amount of nodes, there must be at least one sinks for about every 32 nodes. In all simulations, unless otherwise specified, each network contains two data sinks.

- **Network topology.** Both sensor nodes and data sinks are deployed randomly over the simulation area.
- **Simulation area.** In the random networks, the nodes are deployed in an area of 100 by 100 meters. The nodes and sinks have a transmission range of 16.1 meters.
- **Simulation duration.** The duration of each network simulation run is of 5.000 MAC frames, so each sensor node has 5000 times the opportunity of performing some action i.e. generating and sending data or updating its parent.
- **Number of simulation runs.** Every simulation run is repeated 200 times, after which the results are averaged. In each of the 200 simulation runs, one random network topology is generated and all four algorithms are simulated on that random network topology.
- **Packet rate.** The packet rate is the rate at which nodes generate data packets, which contain sensor readings. Depending on the network structure and the number of nodes and sinks, a too low packet rate means that all algorithms and routing metrics show the same results, while a too high packet rate congests the network too much and no clear results can be obtained from it. Simulation results (Appendix I, Figure 36 and Figure 37) show the influence of the packet rate on the performance of the routing metrics. We will use a typical packet rate, which results in an average packet delivery ratio of about 90%.
- **Routing path update rate.** Nodes update their parent with a chance of 10% per frame.
- **Packet buffer size.** Nodes have a packet buffer able to contain eight packets.
- **LMAC parameters.** Number of timeslots per frame and the degree of the network are closely related to each other. The number of timeslots per frame is set at 16. Therefore, only networks with a maximum degree of 16 are accepted as suitable networks. A timeslot consists of a CM section of 114 bits and a data section of 2040 bits. More information about the frame structure of LMAC can be found in [7].
- **Radio model.** The same radio model as in [16] and [34]. In this model, the transmission (TXC) and receive (RXC) costs are defined as:

$$TXC(k) = \varepsilon_{lec} * k + \varepsilon_{amp} * k * d^2 \quad (4.1)$$

$$RXC(k) = \varepsilon_{lec} * k \quad (4.2)$$

With the parameters as defined in Table 6. At MAC level every node sends one CM section per frame and receives one CM section per frame for every one-hop neighbour it has. When a node transmits a packet to its parent it will send it in its DM section. When a node receives data from its parent, it receives it in the DM section of its parent.

Table 6 – Radio model parameters

Term	Definition	Value	
d	Transmission range	16.1	
k	Number of bits to transmit	**CM section**	**DM section**
		114	2040
ε_{lec}	Energy required by transmitter or receiver in nJ/bit	50	
ε_{amp}	Energy required by transmitter amplifier in pH/bit/m^2	100	

In Figure 22 an example of a random network topology is drawn.

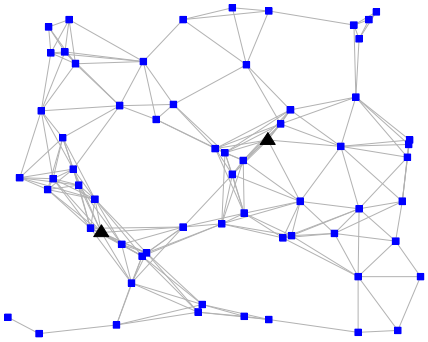


Figure 22 – Random network topology

Three different simulation types

In order to test the performance of the two different routing modes and all the different routing metrics under different network conditions, we will have several different network simulations.

- **Multi-sink performance**. The first simulation type will test the influence of the number of data sinks on the performance of the routing metrics. This is important, because adding multiple sinks to the network should indeed increase the performance of the network. Besides SPR as reference, NCLB and P-NLB's S-SPM and LBM in combination with the *Buffer* routing metric are simulated.
- **Cluster size distribution**. Next, the influence of the initial cluster sizes in the network on the performance of the routing metrics will be tested. Besides SPR as reference, NCLB and P-NLB's S-SPM and LBM in combination with the *Buffer* routing metric are simulated.
- **Routing metric performance**. Finally the performance of all routing metrics in S-SPM and BM, together with NCLB and SPR as reference are compared, using two different network types.

Performance metrics

The performance of the new routing protocol is measured by running network simulations. A random network of 100 nodes is created with four sinks positioned in it. All nodes in the network have a certain chance of generating sensor data packets and forward these packets to their parents until the packets reach a sink. The routing metrics define which parent a node selects from its set of direct neighbours. The performance of the network is measured using the following performance metrics, as discussed in Chapter 1.5:

- Latency
- Network lifetime
- Throughput
- Energy efficiency
- Packet delivery ratio
- Standard deviation of load per sink

In order to keep this chapter short and readable, the rest of the simulation results are included in Appendix I. These results include the standard deviation of each simulation, which gives insight in the variation of the individual simulation runs.

4.2 Simulation Results – Multi-sink performance

As shown in the graphs in Figure 23 all algorithms benefit from an increasing amount of sinks in the network. However, increasing the number of sinks does not scale linear with the increase in performance; each added sink increase the performance a bit less.

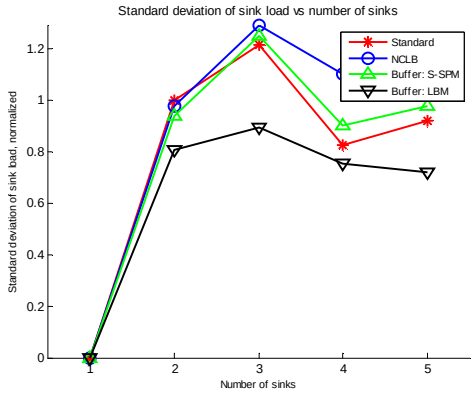
In the first graph is clearly visible that LBM does precisely what it is supposed to do; it is much better able to uniformly distribute the load over all the sinks in the network than the other algorithms. The advantage is the largest with two sinks in the network, but decreases when more sinks are added to the network. The reason for this is that the average hop count between sinks and nodes decreases and there are relatively many sinks close to a sink, when there are more sinks in the network.

The latency results show that the largest increase is by going from one sink to two sinks. NCLB starts with a much lower latency than SPR and both modes of P-NLB, but this difference decreases as the number of sinks increases. S-SPM is slightly better than LBM. Thus, in this graph we notice that balancing the load as done by LBM does not improve the latency.

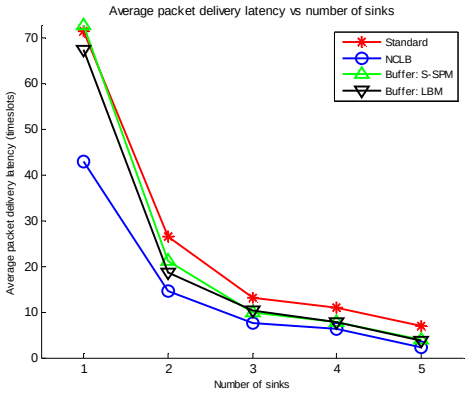
This observation is also visible in the other graphs; although LBM balances the load better over the sinks, throughput and PDR does not benefit from this. One sink in the network is not able to process all traffic load in the network and PDR is quite low, regardless of the used algorithm. With multiple sinks in the network the performance of SPR stays behind of that of the other algorithms. Throughput shows a similar graph, although NCLB achieves a higher throughput.

Network lifetime increases when more sinks are added to the network, although NCLB is better and SPR worse. This logical, since the average routing path length is shorter and thus less energy is consumed on delivering a packet.

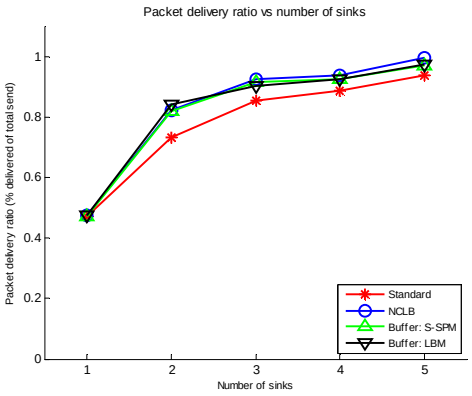
Standard deviations of the graphs of Figure 23 can be found in Appendix I, Figure 28. As shown in those figures, the standard deviation of the latency is quite high, up to 50% of the average latency.



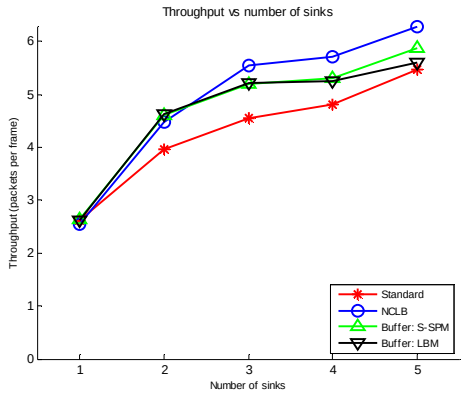
a) Standard deviation of load per sink



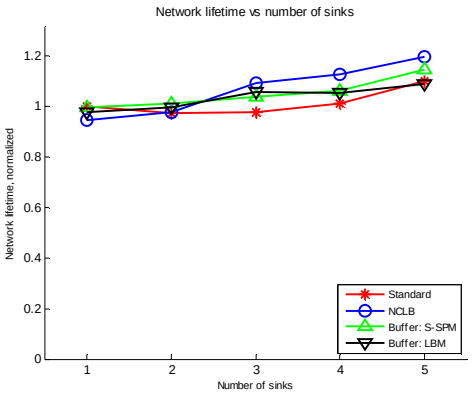
b) Average latency



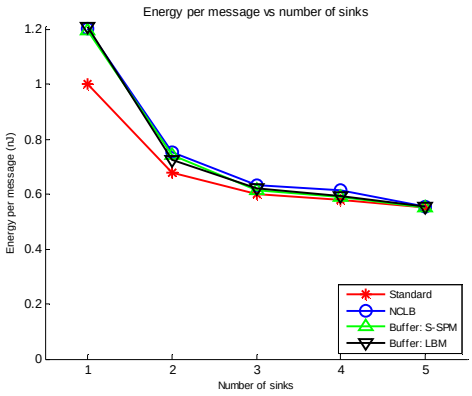
c) PDR



d) Throughput



e) Network lifetime



f) Energy efficiency

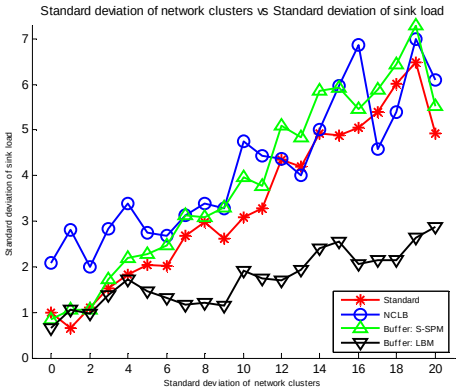
Figure 23 – Multi-sink performance

4.3 Simulation Results – Cluster size distribution

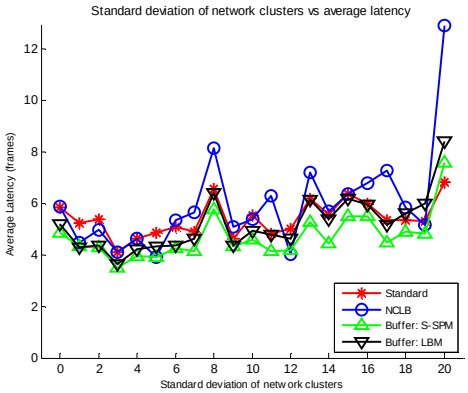
The inter-cluster load balancing technique of P-NLB's LBM should perform best in networks in which the clusters are very different in size. In these simulations the difference between cluster sizes is increased from 0 to 20. This achieved by creating random topologies and determining the difference between the clusters (the standard deviation). Random topologies are created until each standard deviation from 0 to 20 is generated 200 times. From these simulations, shown in Figure 24, can be concluded that LBM is indeed most capable of keeping the load over the sinks uniformly distributed, while the cluster size variance increases. However, we observe again that this leads not to a better result in the other performance metrics.

Latency is lowest using S-SPM and highest using NCLB. LBM performs not as well as S-SPM, but stiller better than SPR and NCLB. The PDR of SPR is clearly worse than that of the other three algorithms, of which the results lay close together. NCLB has the highest PDR, although the difference with P-NLB is only a few percent. The throughput graphs show similar results, except that NCLB is now distinguishably better than both modes of P-NLB. By using NCLB the highest network lifetime is achieved, SPR performance is worse and LBM slightly better than S-SPM. Energy efficiency is about the same for all algorithms, although S-SPM has slightly better results than the other three algorithms.

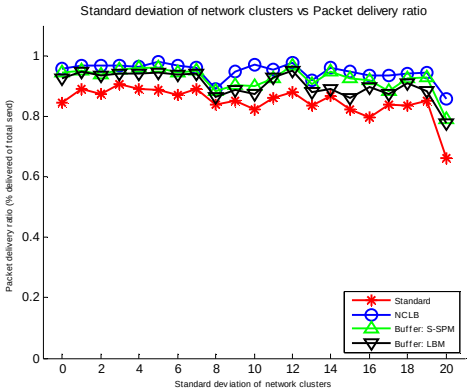
Generally speaking, the performance of LBM is not as expected. Although it is for all six performance metrics better than SPR, for most performance metrics the performance of LBM is worse than that of S-SPM. In Section 4.5 an elaborate discussion about explanations for the bad performance of LBM can be found. In that Section 4.5 discusses also the standard deviations of the graphs of Figure 24. As can be seen in Figure 29, these standard deviations are quite high, up to 100 % of the average value.



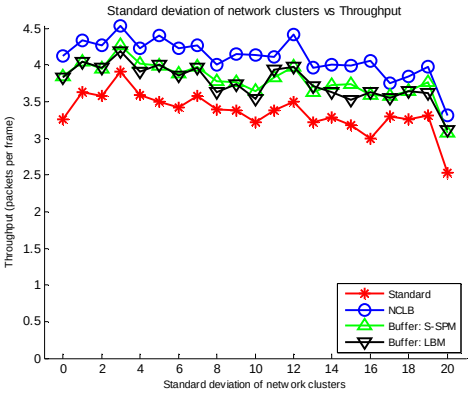
a) Standard deviation of load per sink



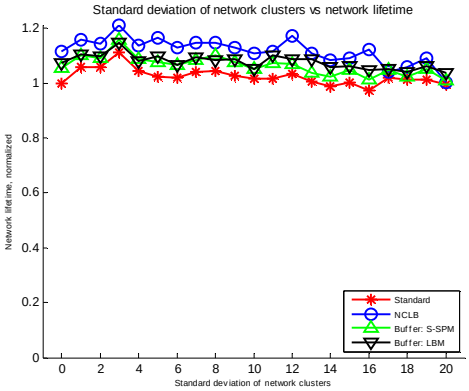
b) Average latency



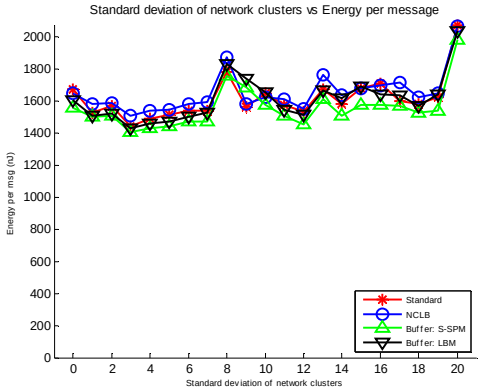
c) Packet delivery ratio



d) Throughput



e) Network lifetime



f) Energy efficiency

Figure 24 – Simulation results of the cluster size distribution

4.4 Simulation Results – Routing metric performance

Routing Metrics

All routing metrics have already been discussed in Section 3.4.1. The routing metrics are used in the two different modes of P-NLB: S-SPM and LBM.

- Free space in receive buffers
- Number of child nodes
- Number of upstream nodes
- Energy level of nodes

Two different network structures

While previously simulations are only run on the random network topology the simulations of the routing metric performance are also done on another network type: a network with two clusters with different sizes:

- Network consisting of two connected clusters of unequal size. One cluster has 25 nodes, the other has 40 nodes, both clusters contain one sink – the difference is thus 15 nodes. Test setup should prove a load balanced network is an efficient network. This network type is hereafter called **asymmetric clusters**.

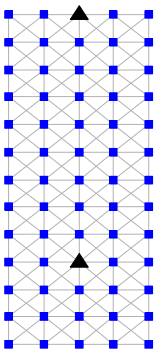


Figure 25 – Symmetric cluster topology

4.4.1 Routing metric performance: Random topology

When looking at the performance of all algorithms in random topologies in Figure 1, LBM is still better in distributing the load over the sinks. All routing metrics have more or less the same load for each routing mode.

NCLB results in the lowest latency in the random topology, 50% lower than SPR. Latency is in general higher in LBM than in S-SPM. Latency is lowest when using routing metrics *Buffer*, in both routing modes. In S-SPM mode the latency comes close to the latency of NCLB. In LBM the latency with using routing metric *Buffer* is lower than three routing metrics of S-SPM.

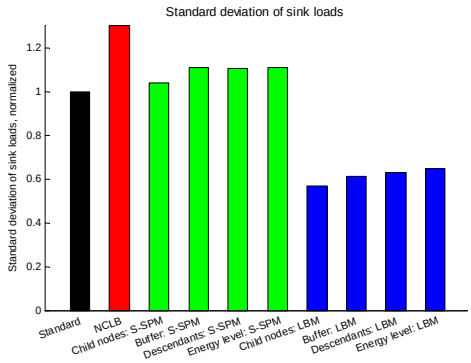
PDR is highest when using *Buffer* as metric in S-SPM. NCLB has a higher PDR than all other algorithms, although the results do not differ very much. LBM stays behind, with a PDR equal to that of SPR.

The throughput of SPR is lowest of all algorithms, NCLB and routing metric *Network lifetime* in S-SPM results in the highest throughput. The throughput of LBM is up to 10% worse than that of S-SPM.

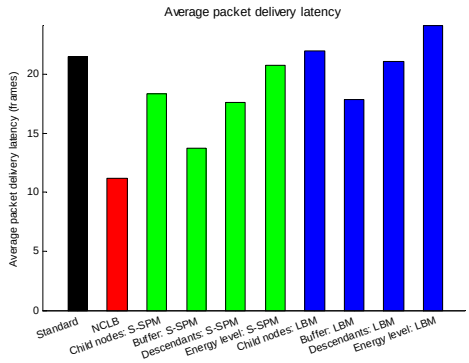
Logically, routing metric *Network lifetime* results in the highest network life, in S-SPM up to 10% higher than SPR and NCLB. The energy efficiency shows no great differences.

The standard deviation of these graphs can be found in Appendix I, Figure 30. Again these standard deviations are very high.

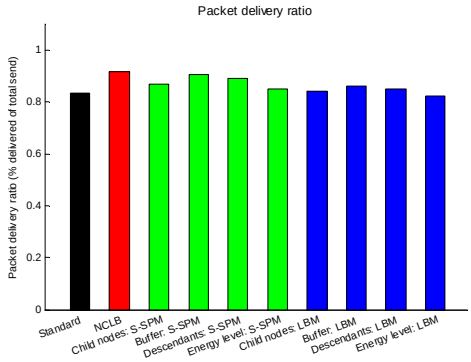
In general latency varies most among the different routing metrics, where routing metric *Buffer* performs clearly the best in both routing modes. By using this routing metric, nodes are best in avoiding congestion and consequently this results in the lowest latency and highest PDR. The results of the other performance metrics show more or less equal results for all routing metrics, although routing metric *Network lifetime* achieves a better throughput and network lifetime. Looking at all results; NCLB performs in general the best, with lowest latency and highest PDR. In almost all cases SPR performs worse than all other algorithms and LBM worse than S-SPM.



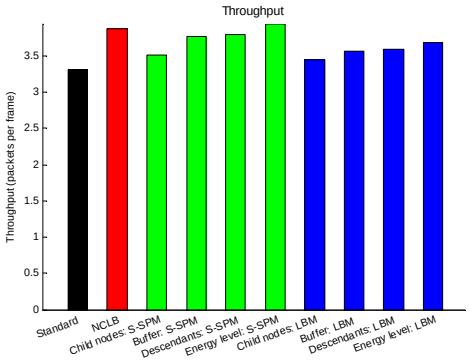
a) Standard deviation of load per sink



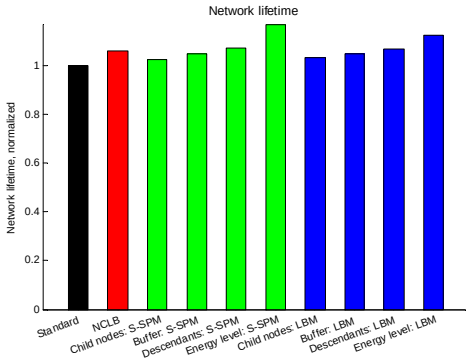
b) Latency



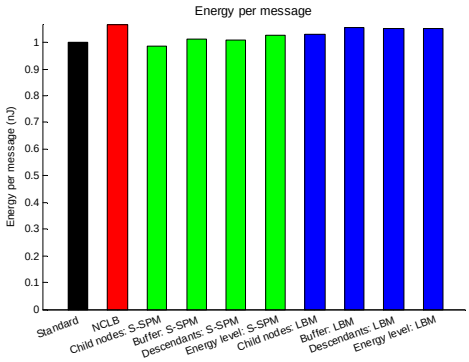
c) PDR



d) Throughput



e) Network lifetime



f) Energy efficiency

Figure 26 – Simulation results random topology

4.4.2 Routing metric performance: Asymmetric clusters topology

The asymmetric clusters topology shows again that LBM is able to distribute the load uniformly over the sinks. This results in a much lower latency for the routing metrics *Child nodes*, *Buffer* and *Descendants* using LBM compared with using S-SPM, and even lower than NCLB. The decrease in latency compared with SPR is more than 50%. Routing metric *Energy level* has in both S-SPM and LBM a higher latency, probably caused by the long routing path in order to avoid nearly depleted nodes around the sink. The lowest latency of all algorithms is achieved by using routing metrics *Buffer* and *Descendants* in LBM.

If routing mode LBM is used, PDR is up to 10% higher than all other algorithms. The highest PDR is also achieved by using routing metrics *Buffer* and *Descendants* in LBM. Throughput is showing other results than the PDR. This can be explained by the fact that traffic load on the top-level neighbours of the sinks is more important for the throughput than up to those nodes. Therefore, it is no surprise the routing metrics *Buffer*, *Descendants* and *Energy Level*, which can best route around congested top-level neighbours to less congested top-level neighbours, have the highest throughput. Of course, NCLB has the highest throughput, since that protocol makes the best use of the top-level neighbours of the sinks.

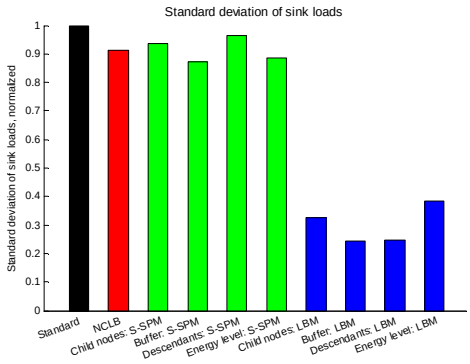
As expected, the network lifetime, is highest when using the routing metric *Network lifetime*. NCLB also results in a high network lifetime, because it distributes the load over all neighbours of the sinks. Since these neighbours are likely to run out of energy first, this approach extends the lifetime of those nodes. The lifetime of the whole network is increased up to 10%, in comparison with SPR.

The energy efficiency performance metrics show that the long routing paths of routing metric *Network lifetime* result in relatively much energy is used to deliver the packets at the sinks.

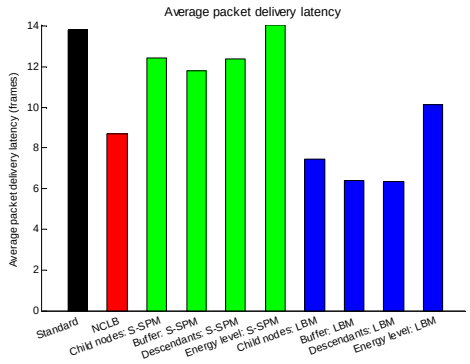
The standard deviations of these graphs, which can be found in Appendix I, Figure 31, are quite low.

In general it is obvious that routing metric *Child nodes* performs worst of all routing metrics in this regular network type, since most nodes have an equal degree. Therefore, this metric cannot gain any advantage. Routing metric *Buffer* is most suitable if a low latency or high PDR are needed. Routing metric *Network lifetime* can best be used if a high network lifetime or throughput is required. In this network type, P-NLB is able to outperform the centralized algorithm of NCLB in performance metrics

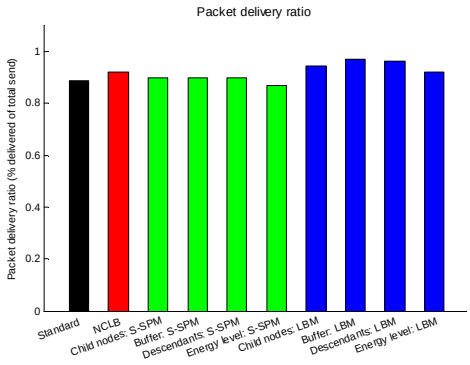
latency in PDR and achieve the same performance in performance metrics network lifetime and throughput.



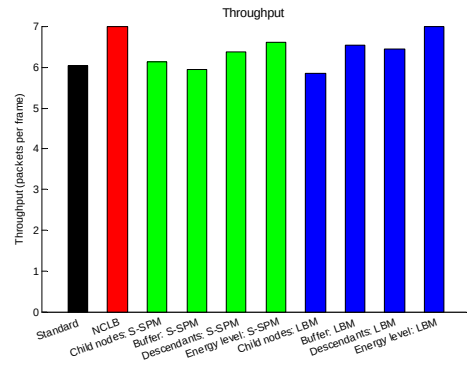
a) Standard deviation of load per sink



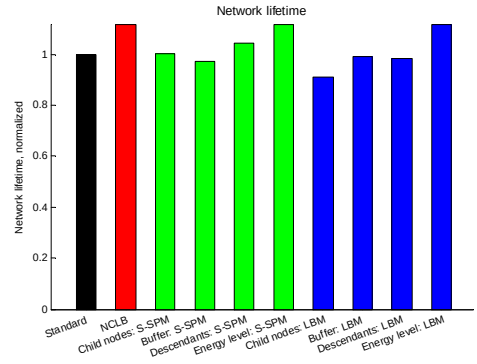
b) Latency



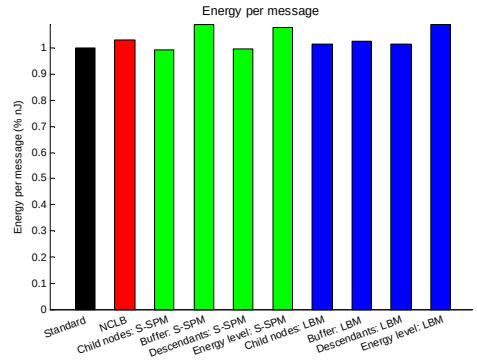
c) PDR



d) Throughput



e) Network lifetime



f) Energy efficiency

Figure 27 – Simulation results two non-uniform clusters topology

4.5 Evaluation of all simulation results

With all the collected simulations results, one can clearly see that the load balancing mechanism of LBM does balance the load more equally over all the sinks in the network. However, in the random network type this has not the expected positive effect on the other performance metrics. In the multi-sink simulations, LBM performs more or less equal to S-SPM. Simulations where the difference between cluster sizes is increased step-by-step shows that the performance of LBM is even slightly worse when compared with S-SPM. The last simulation sequence shows that most routing metrics perform worse in LBM than in S-SPM. Exceptions are routing metrics *Buffer* and *Network lifetime* which are able to achieve a fair latency and network lifetime. In a regular topology with two clusters of different sizes, such as the asymmetric clusters topology, the benefits of load balancing are much more obvious. Latency, PDR, and throughput are between 10% and 40% better. When looking at application targets, the conclusion can be drawn that routing metric *Buffer* leads to the lowest latency and highest PDR, while routing metric *Network lifetime* results in the highest throughput and longest network lifetime.

Although the load in the network is more balanced using balancing mode, we have seen that latency, PDR and throughput do not reflect this. Reasons for that are:

- Incidentally created loops cause temporary extra latency
- Longer path lengths cause extra latency
- Sinks are in most cases not the bottleneck in the network, but congestion occurs sooner in nodes around the sinks – the *top-level nodes*. Actually a sink is in some WSN anything but a bottleneck, because as the terminal station, it has much more bandwidth to the end-user (network) and is able to forward nodes faster from its buffer. Therefore load balancing the load over the sinks does not always lead to a better performance.
- Local bottlenecks caused due to irregular structure of the random topology networks have in some WSN more influence on the performance, than the load on the sinks.
- Bandwidth distribution. By using the TDMA-based LMAC as underlying MAC protocol, its collision free scheduling technique has as drawback in fixed but reduced bandwidth per node. Every node can send only one packet per frame, no matter if it has many packets to send – when the node is congested – or that is has no packets in its message queue. This increases the negative effect that bottlenecks in the network have on the performance of the protocol. This also limits the maximum throughput in the network; the maximum throughput is equal to the number of top-level neighbours of the sinks. Contention-based protocols like CSMA, might be better in reducing congestion, assuming the density of the nodes is not too high.

The centralized NCLB algorithm performs in almost all cases better than the distributed P-NLB. Surprisingly, in some cases P-NLB is still able to outperform NCLB, for example the latency of both S-SPM and LBM are better in the cluster size distribution metrics and LBM outperforms NCLB in most performance metrics in the asymmetric clusters simulations. The higher latency is caused by the (much) longer routings path as a result of NCLB's balancing mechanism. The main source of the better performance of NCLB is the cardinality of all top-level branches of the routing trees in the networks. LBM tries the balance the load of all the whole routing trees (clusters) in the network, while NCLB tries to balance the load of each top-level branch in the routing trees. Since the sinks have better processing capabilities than common sensor nodes, these sinks are in many cases not the bottleneck in the network, but the top-level nodes are. Therefore balancing the load in those top-level branches proves to be more effective. Unfortunately this is much harder to achieve and therefore requires a centralized method. Nevertheless investigating in a distributed mechanism for LBM which is better to balance the load in those top-level branches could produce good results.

A point of discussion is the standard deviation of the simulations results. The standard deviations of the simulation results of the random network topology are quite high, up to 20% of the average value. The source for this is the great variation of random network topologies, causing networks with short and long path length. Also, the average and variation of the degree in the networks is quite high due to the random deployment. These variations result in great differences between the performances of a sequence of simulation runs. So, although the number of simulation runs is quite high, the standard deviation is still high. Experiments where the number of runs is increased up to 500 show no decrease in the standard deviation.

Finally, all simulation results show that P-NLB, in both modes, outperforms SPR in all performance metrics. The metric-based tree building of S-SPM and the addition of inter-cluster load balancing in LBM is the source for that improved performance.

Chapter 5

5. Conclusion

This thesis presented P-NLB, a routing protocol for large-scale multi-sink WSN, with uses global clustering with inter-cluster load balancing technique in combination with local metric-based routing for optimized routing tree building. On the global level information about cluster sizes in the network is gathered by sinks and distributed to the sensors. Its distributed approach, in which each node autonomously decides what best routing path is, results in very low communication overhead due to the use of cross-layer information of the MAC layer and flexibility of the routing trees. Except for a one-time broadcasting for detecting initial cluster sizes, as part of the LMAC setup phase, only local information exchange is used, making it very scalable to large sensor networks. Finally it requires no geographical location information of nodes at all.

Simulations show that P-NLB's LBM uniformly distributes the load efficiently over the sinks in the network. In random network topologies this results in a higher latency, caused by longer routing paths. Packet delivery ratio does not always benefit from balancing the load; LBM gains most advantage in comparison with shortest path mode, when the initial difference between clusters' sizes increases. Both routing modes of P-NLB outperform SPR in all simulations. NCLB achieves the highest performance in most simulations, which is no surprise since it is a centralized algorithm. Evaluation of the four defined routing metrics show that using routing metric *Buffer* leads to the lowest latency and highest PDR. When the application target is a long network lifetime or high throughput, using routing metric *Network lifetime* leads to the best results.

Although the load in the network is more balanced using LBM, latency and PDR does not reflect this. Sources for that are:

- Incidental created loops cause extra packet latency
- Longer path length causes extra packet latency
- Sinks are in most cases not the bottleneck in the network, but congestion is more likely to appear in nodes around the sinks, since the traffic load in the network converges to those nodes.
- Local bottlenecks have great negative influence on performance.
- Fixed, but limited bandwidth per node, due to use of TDMA based LMAC as underlying MAC.

NCLB is much better in tackling this problem, since it balances the load in each top-level branch in the spanning trees. It has a much better PDR, although it latency increases due to the longer routing paths. NCLB however, is completely centralized and therefore not complying to the demands of this thesis, i.e. flexible, decentralized solution.

5.1 Future work

Mobility

Mobility of nodes and sinks in the network can create many problems in routing path construction. Many related works are specialized in the topic of mobility in WSN [40, 41]. Due to the flexibility of P-NLB in constructing routing spanning trees and the lack of explicit maintenance of them in combination with the decentralized approach, P-NLB is assumed to be very suitable for mobile WSN. However, further research is needed to evaluate the performance of P-NLB in mobile WSN.

Message Priority

In some WSNs different message types can be distinguished with different priorities, i.e. periodic low priority sensor status data, infrequent high priority even information. It would be useful if the routing protocol could give different routing priorities to different message types. A high priority message would be delivered as soon as possible to a data sink, while low priority messages would be route as energy efficient as possible in order to prolong network lifetime. A technique for achieving this could be creating several ‘virtual’ spanning trees in the network, each used for routing message with a different priority. One spanning tree can be focussed on latency, other on throughput and the last one on energy efficiency.

Experimental verification of simulation results

In order to test the performance of the protocol in a real test-bed, experiments must be done. Therefore, the protocol must be implemented on sensor nodes, for example on Ambient μNodes [39]. A small scale test with a few sensors should be able to show if the simulation results of the protocol, also hold on hardware.

Bibliography

[1] Woo, A., Tong, T., Culler, D.: Taming the Underlying Challenges of Reliable multihop Routing in Sensor Networks. In: Proceedings of the 1st international conference on Embedded networked sensor systems, pp. 14--27 (2003)
[2] Lee, J.J., Krishnamachari, B., Kuo, J.: Impact of Energy Depletion and Reliability on Wireless Sensor Network Connectivity. In: Digital wireless communications Conference 6, Vol. 5440, pp. 169--180 (2004)
[3] Pei, G., Chien, C.: Low Power TDMA in Large Wireless Sensor Networks. MILCOM 2001 - IEEE Military Communications Conference, no. 1, October 2001, pp. 347—351
[4] Wang, L., Xiao, Y.: A survey of energy-efficient scheduling mechanisms in sensor networks. In: Mobile Networks and Applications, Volume 11, Issue 5 (October 2006), pp. 723—740, Kluwer Academic Publishers (2006)
[5] Younis, M., Youssef, M., Arisha, K.: Energy-Aware Routing in Cluster-Based Sensor Networks. In: Proceedings of the 10th IEEE Int.l Symp. on Modeling, Analysis, & Simulation of Computer & Telecommunications Systems (MASCOTS.02), pp. 129-136
[6] Zhou, C., Krishnamachari, B.: Localized Topology Generation Mechanisms for Wireless Sensor Networks. In: IEEE GLOBECOM, pp. 1269—1273. (2003)
[7] Hoesel, L.F.W., Havinga, P.J.M.: Design Aspects of An Energy-Efficient, Lightweight Medium Access Control Protocol for Wireless Sensor Networks. In: Int. J. Commun. Syst. 2006, pp. 1—21
[8] http://www.aware-project.net
[9] Xuan, H.L., Lee, Y., Lee, S.: Two Energy-Efficient Routing Algorithms for Wireless Sensor Networks. In: Lecture Notes in Computer Science, Networking - ICN 2005, Vol. 3420/2005, Springer Berlin / Heidelberg (2005)
[10] Lukošius, A.: Opportunistic Routing in multi-sink mobile ad hoc wireless sensor networks, Master Thesis (2007)
[11] Ciciriello, P., Mottola, L., Picco, G.P.: Efficient Routing from Multiple Sources to Multiple Sinks in Wireless Sensor Networks. In Proceedings of the 4th European Conference on Wireless Sensor Networks (EWSN 2007), Delft (The Netherlands), January 2007.
[12]Tan, H.P., Gabor, A.F.m Seah, W.K.G., Lee, P.W.Q.: Performance Analysis of Data Delivery Schemes for a Multi-sink Wireless Sensor Network. 22nd International Conference on Advanced Information Networking and Applications 2008, pp. 418—425 (2008)
[13] Oyman, E.I., Ersoy, C.: Multiple Sink Network Design Problem in Large Scale Wireless Sensor Networks. In: IEEE International Conference on Communications 2004, Vol. 6, pp. 3663-3667 (2004)
[14] Ahmed, A.A., Fisal, N.: A real-time routing protocol with load distribution in wireless sensor networks. In: Computer Communications Vol. 31, Issue 14, pp. 3190—3203 (2008)
[15]Puccinelli, D., Haenggie, M.: Network-Layer Load Balancing for Wireless Sensor Networks. In: IEEE Wireless Communications and Networking Conference 2008, pp. 2063—2068 (2008)
[16]Heinzelman, W.R., Chandrakasan, A., Balakrishnan, H.: Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In: Proceeding of the 33rd Annual Hawaii International Conference on System Sciences, pp. 1--10 (2000)
[17] Bejerano, Y., Han, S.J., Kumar, A.: Efficient Load-Balancing Routing for Wireless Mesh Networks. In: Computer Networks, Vol. 51, No. 10, pp. 2450--2466 (2007)

[18] J. M. Mendel, "Fuzzy Logic Systems for Engineering: A Proceedings of the IEEE vol. 83 no. 3 Tutorial", March 1995, pp. 345-377.
[19]Oyman, E.I., Ersoy, C.: Multiple Sink Network Design Problem in Large Scale Wireless Sensor Networks. In: Proceedings of the International Conference on Communications Volume 6, pp. 3663--3667 (2004)
[20] Kim, H., Seok, Y., Choi, N., Choi, Y., Kwon, T.: Optimal Multi-sink Positioning and Energy-efficient Routing in Wireless Sensor Networks. In: Lecture Notes in Computer Science, vol. 3391 no. pp.264-274, Jan. 2005.
[21] Low, C.P., Fang, C., Ng, J.M., Ang, Y.H.: Efficient Load-Balanced Clustering Algorithms for wireless sensor networks. In IEEE ICC, pp. 3485—3490 (2007)
[22] Gupta, G., Younis, M.: Load-balanced clustering of wireless sensor networks. In: Proceedings of the IEEE International Conference on Communication, Vol. 3, pp. 1848--1852. May (2003)
[23] Zhang, Y., Huang, Q.: A Learning-based Adaptive Routing Tree for Wireless Sensor Networks. In: Journal of Communications, Vol. 1, No. 2, pp. 12--21. Academy Publisher (2006)
[24]Egorova-Förster, A., Murphy, A.L.: Exploiting Reinforcement Learning for Multiple Sink Routing in WSNs. In: IEEE MASS 2007, pp 1--3 (2007)
[25] Servetto, S.D., Barrenechea, G.: Constrained Random Walks on Random Graphs: Routing Algorithms for Large Scale Wireless Sensor Networks. In: Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications, PP. 12--21. (2002)
[25] Ganesan, D., Govindan, R., Shenker, S., Estrin, D.: Highly resilient, energy efficient multipath routing in wireless sensor networks. In: ACM SIGMOBILE Mobile Computing and Communications Review, v.5 n.4, October 2001
[26] Karlof, C., Li, Y., Polastre, J.: Arrive: Algorithm for robust routing in volatile environments. EECS Department, University of California, Berkeley, UCB/CSD-03-1233 (2003)
[27] Hefeeda, M.: Forest Fire Modeling and Early Detection using Wireless sensor networks. In: Technical Report TR 2007-08, School of Computing Science, Simon Fraser University, August 2007.
[28] Deng, J., Han, R., Mishra, S.: A Robust and Light-Weight Routing Mechanism for Wireless Sensor Networks. In: 1st Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (2004)
[30] Chatterjee, P., Das, N.: A Distributed Algorithm for Load-Balanced Routing in multihop wireless sensor networks. In: 9th ICDCN, LNCS 4904, pp. 332—338. (2008)
[31] Dai, H., Han, R.: A Node-Centric Load Balancing Algorithm for wireless sensor networks. In: IEEE GLOBECOM, pp. 548—552. (2003)
[32] Gao, J., Zhang, L.: Load Balanced Short Path Routing in Wireless Networks. In: IEEE INFOCOM, pp. 1099--1108. (2004)
[33] Raicu, I., Schwiebert, L., Fowler, S., Gupta, S. K.S.: Local Load Balancing for Globally Efficient routing in wireless sensor networks. In: International Journal of Distributed Sensor Networks, Vol. 1, pp. 163–185. Taylor & Francis Inc. (2005)
[34] Azim, M.A., Jamalipour, A.: Optimized Forwarding for Wireless Sensor Networks by Fuzzy Inference System. In: IEEE International Conference on Wireless Broadband and Ultra Wideband Communications, Sydney, 13-16 March 2006
[35] Perkins, C.E., Royer, E.M.: Ad-hoc on-demand distance vector routing. In: Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, pp. 90-100. (1999)
[36] Mueller, S., Tsang, R.P., Ghosal, D.: Multipath routing in mobile ad hoc networks: issues and challenges In: LNCS Vol. 2965 pp. 209–234. (2004)
[37] Sun, P., Seah, W.K.G., Lee, P.W.Q.: Efficient data delivery with packet cloning for underwater sensor networks. In: Proc. of the Intl. Symp on Underwater Technology, pp. 34–41. (2007)
[38] Xie, P., Cui, J.H., Lao, L.: Vector-Based Forwarding Protocol for Underwater Sensor Networks. In: Proceedings of Networking, pp. 1216–1221. (2006)
[39] http://www.ambient-systems.net
[40]Wu, J., Havinga, P.J.M: Reliable Cost-based Data-centric Routing Protocol for Wireless Sensor Networks. In: Proceedings of the Seventh ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD'06), pp. 267—272
[41]Puccinelli, D., Brennan, M., Haenggi, M.: Reactive Sink Mobility in Wireless Sensor Networks. In: Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking, pp. 25--32 (2007)

List of Abbreviations

Abbreviation	Full phrase
WSN	Wireless Sensor Network
S-SPM	Smart Shortest Path Mode
LMAC	Light-weighted Medium Access Control
LBM	Load Balancing Mode
MAC	Medium Access Control
LLC	Logical Link Control
QoS	Quality of Service
ART	Adaptive Routing Tree
P-NLB	Partition-based Network Load Balancing
SPR	Shortest Path Routing
MST	Minimum Spanning Tree
PDR	Packet Delivery Ratio
PACT	Power Aware Clustered TDMA
TDMA	Time Division Multiple Access
SDMA	Space Division Multiple Access
CM	Control Message
DM	Data Message
FROMS	Feedback Routing for Optimizing Multiple Sinks
RSSI	Received Signal Strength Indicator
LEACH	Low-Energy Adaptive Clustering Hierarchy
E3D	Energy Efficient Distributed Dynamic Diffusion
LBC	Load Balanced Clustering
GLBCA	Greedy Load-Balanced Clustering Algorithm
DLBR	Distributed algorithm for Load-Balanced routing
NCLB	Node-Centric Load Balancing
LBSP	Load Balanced Short Path routing
OFFIS	Optimized Forwarding by Fuzzy Inference Systems
RTLD	Real-time routing protocol with Load Distribution
ART	Adaptive Routing Tree
LT	Localized Topology generation mechanisms
AODV	Ad-hoc On Demand Distance Vector routing
MHC	MAC Hop Count
RHC	Routing Hop Count

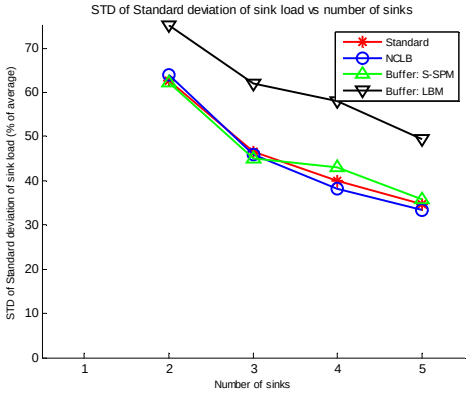
Appendix I

Appendix I contains additional simulation results. The graphs of the following simulations are present in this Appendix:

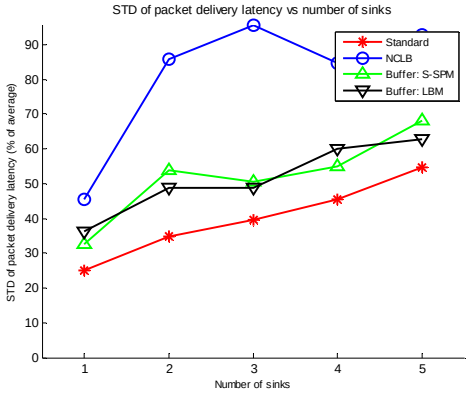
- Standard deviation of simulations in Chapter 4
- Comparison of protocol performance as function of ERP length parameter
- Comparison of protocol performance as function of parent update rate
- Effect of shortest path relaxation on routing metrics Buffer and Energy level
- Comparison of protocol performance as function of packet rate
- Scalability performance by varying the number of nodes and sinks
- Fairness of packet delivery ratio in three different network topologies

Standard deviation of simulation in Chapter 4

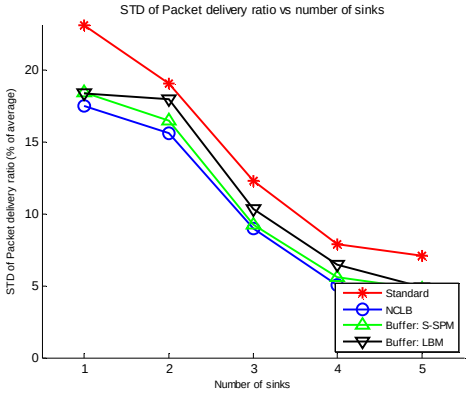
Each simulation consists of 200 runs. In the graphs in Chapter 4 the average values of these 200 runs are taken. The standard deviation (STD) of these 200 runs can be found in this appendix. So the STD in Figure 28, Figure 29, Figure 30 and Figure 31 show the variation between the 200 runs of each simulation. The STDs are not included in the graphs of Chapter 4, since the large STD values would decrease the readability of the graphs in Chapter 4.



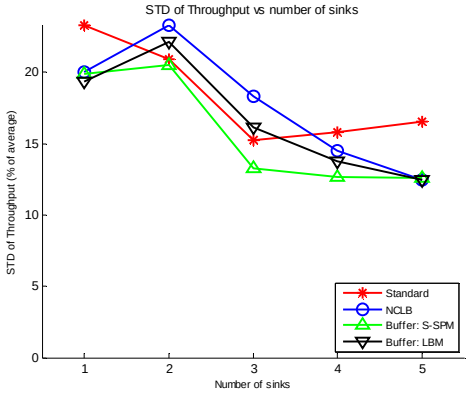
a) STD of standard deviation of load per sinks



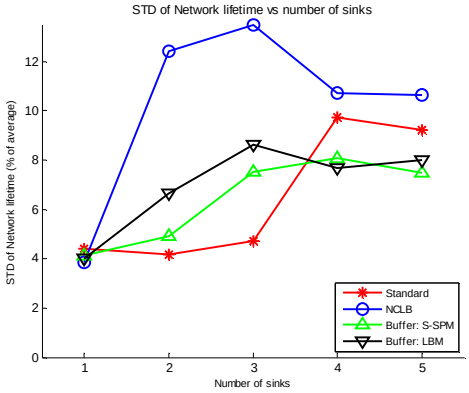
b) STD of latency



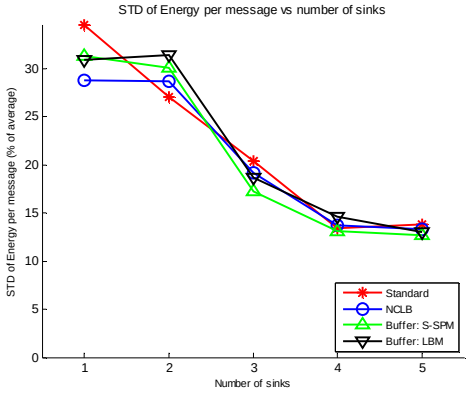
c) STD of PDR



d) STD of throughput

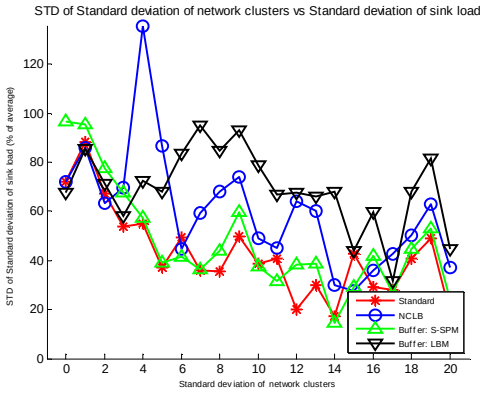


e) STD of network lifetime

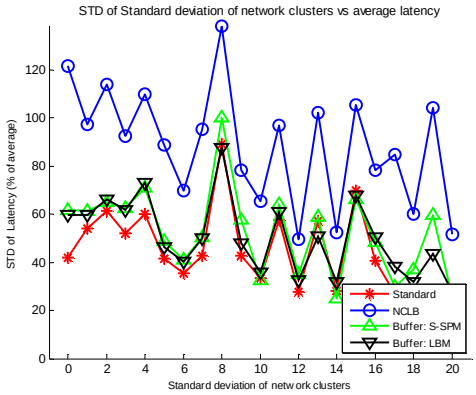


f) STD of energy efficiency

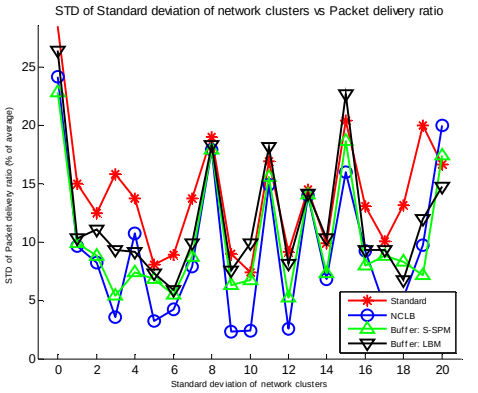
Figure 28 – Standard deviations of multi-sink simulations



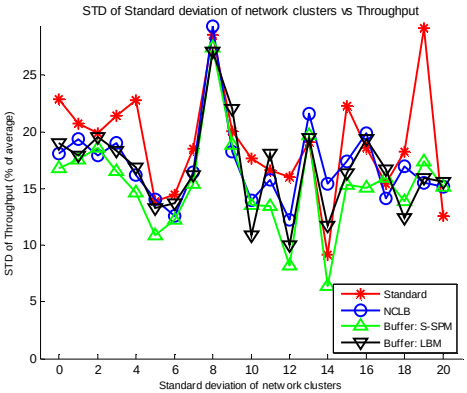
a) STD of standard deviation of load per sinks



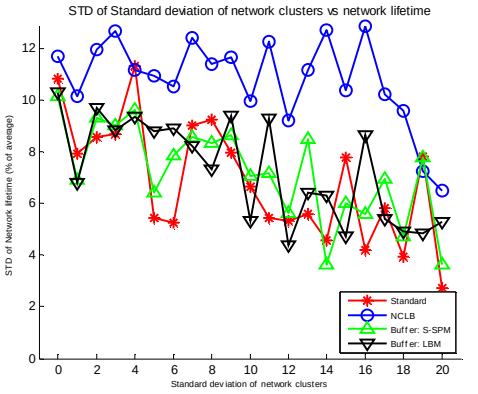
b) STD of latency



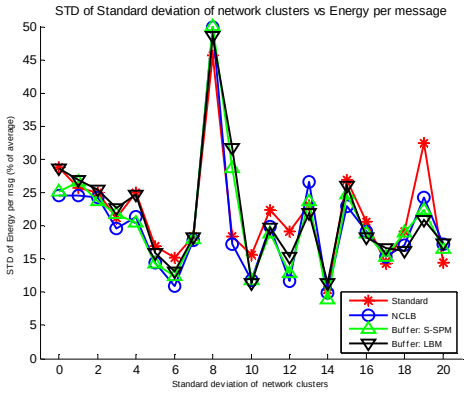
c) STD of PDR



d) STD of throughput

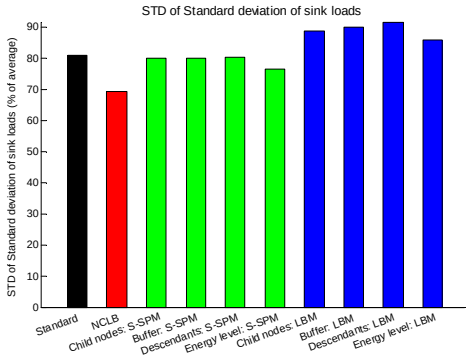


e) STD of network lifetime

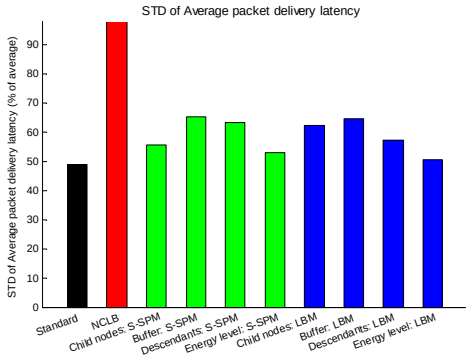


f) STD of energy efficiency

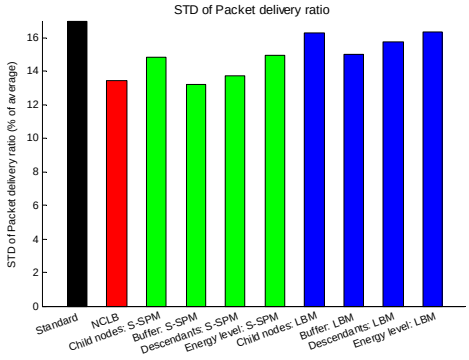
Figure 29 – Standard deviations of cluster size distribution simulations



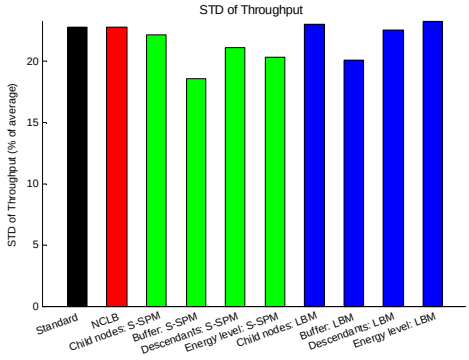
a) STD of standard deviation of load per sinks



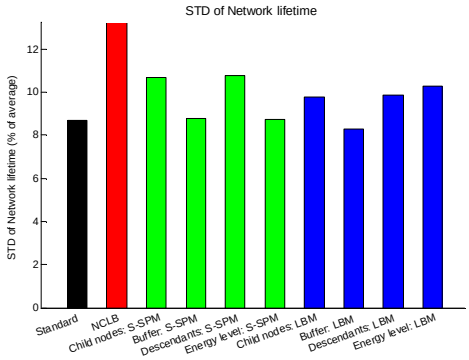
b) STD of latency



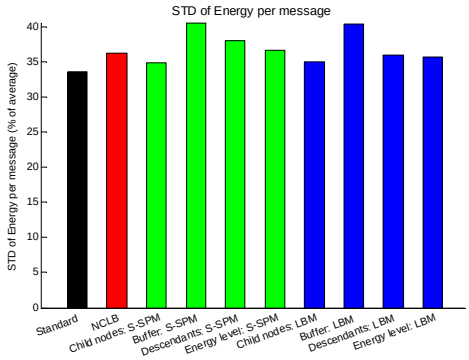
c) STD of PDR



d) STD of throughput

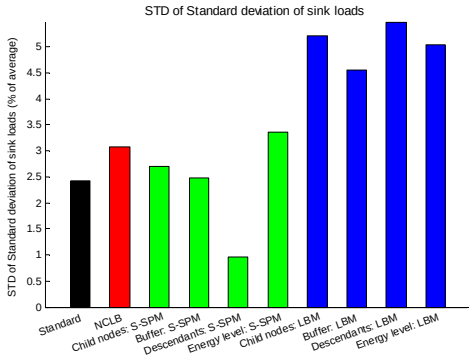


e) STD of network lifetime

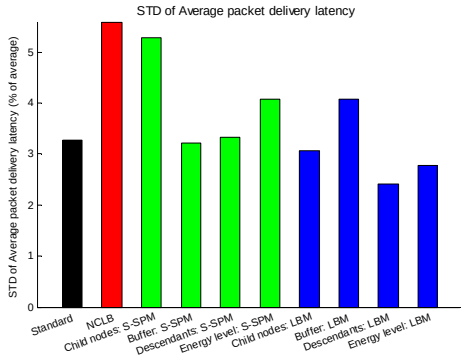


f) STD of energy efficiency

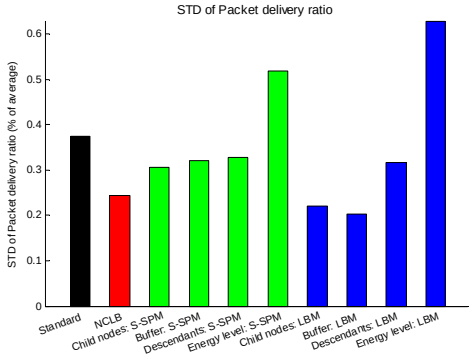
Figure 30 – Standard deviations of routing metric simulations of random network topology



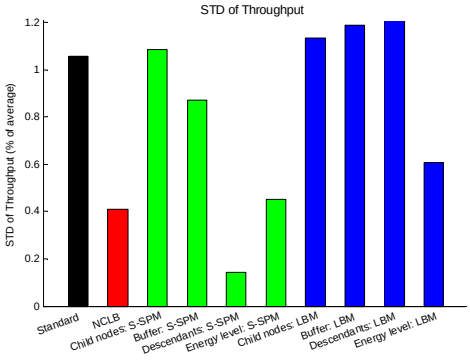
a) STD of standard deviation of load per sinks



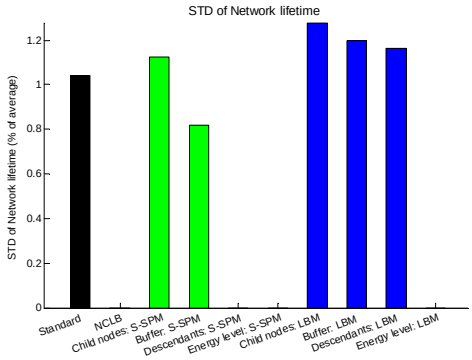
b) STD of latency



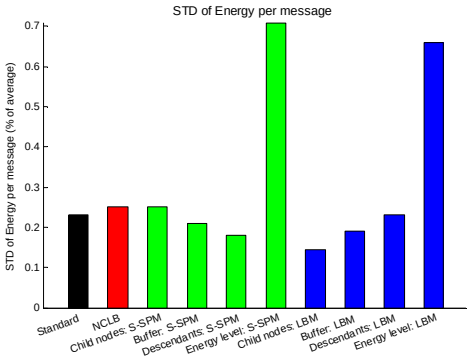
c) STD of PDR



d) STD of throughput



e) STD of network lifetime

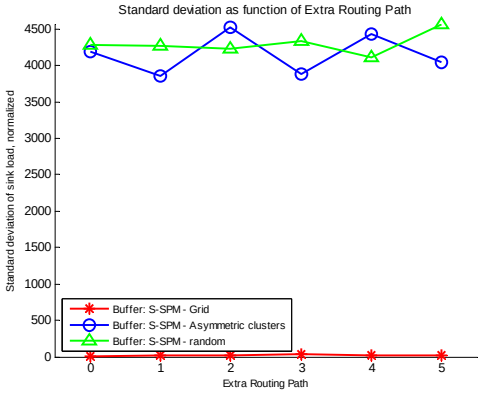


f) STD of energy efficiency

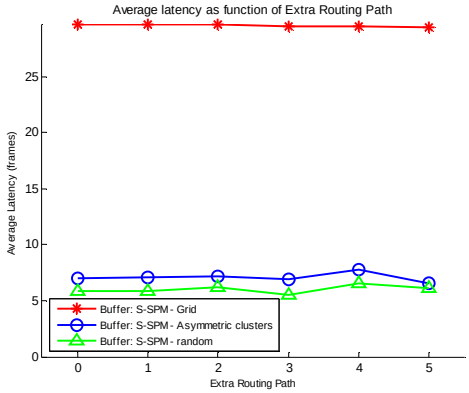
Figure 31 – Standard deviations of routing metric simulations of asymmetric clusters topology

Comparison of protocol performance as function of ERP length parameter

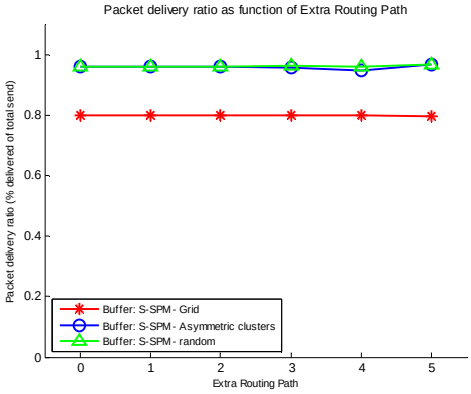
As shown in Figure 35 leads the opportunity of routing to sinks which are at a greater distance initially to an improvement of performance. The standard deviation of the sink load decreases with a minimum at an Extra Routing Path (ERP) of 2, 3 and 4, but increases after that. The latency decreases first if ERP increases to 2, since the load balancing mechanism is able to redirect packets from the congested cluster to the less congested area. With higher ERP the performance gain turns into a performance hit since the longer paths become unstable and the extra hops in the routing tree causes more extra latency than the reduced congestion can compensate for. PDR, Throughput and Energy efficiency show equal values with an ERP of 0 to 4, but with higher ERP the performances decreases, again due to unstable, fast changing routing paths. Network lifetime is more or less equal with all values of ERP.



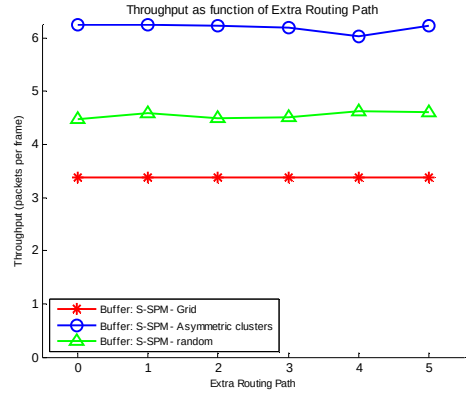
a) Standard deviation of load per sink



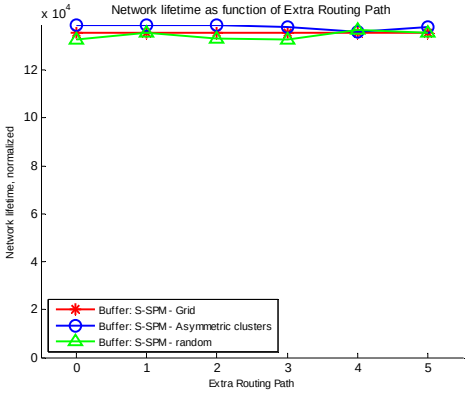
b) Latency

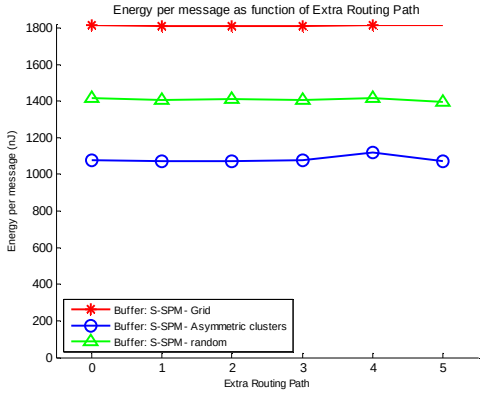


c) PDR



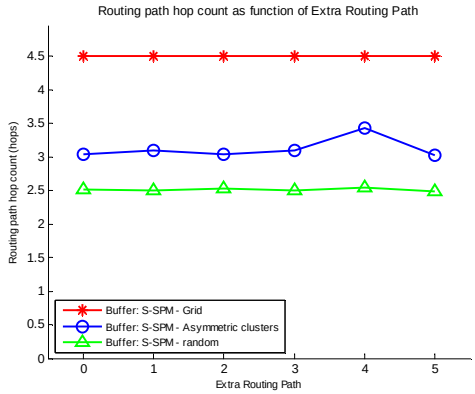
d) Throughput





e) Network lifetime

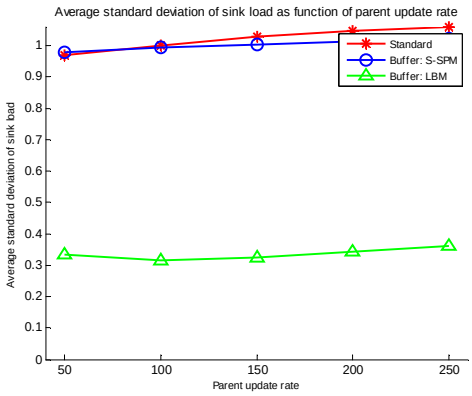
f) Energy efficiency



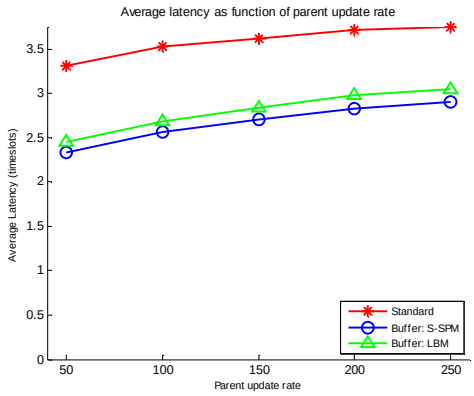
g) Routing path length

Figure 32 – Comparison of protocol performance as function of extra routing path length parameter in all three networks

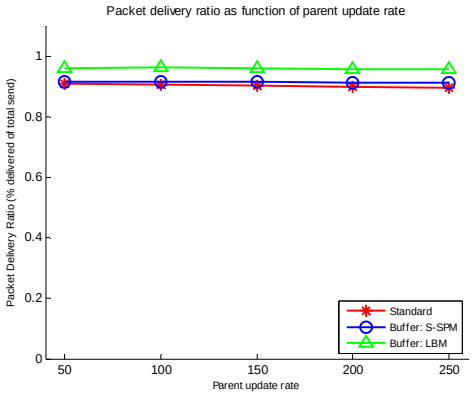
Comparison of protocol performance as function of parent update rate



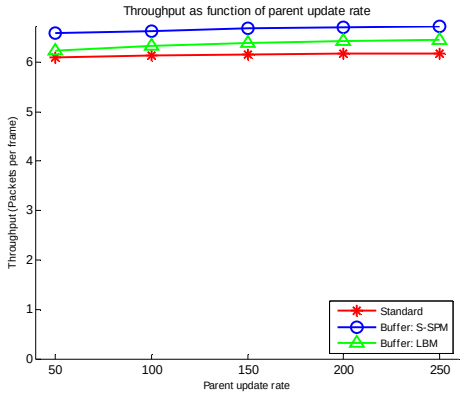
a) Standard deviation of load per sink



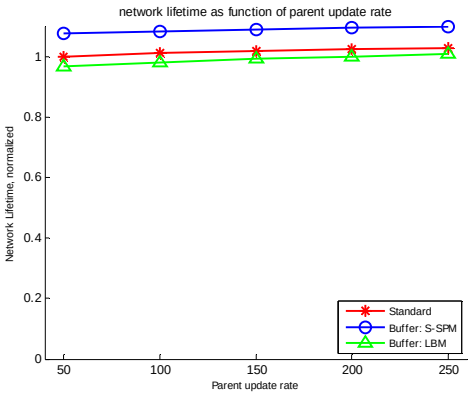
b) Latency



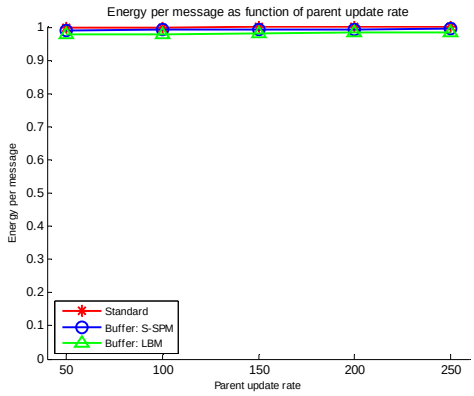
c) PDR



d) Throughput

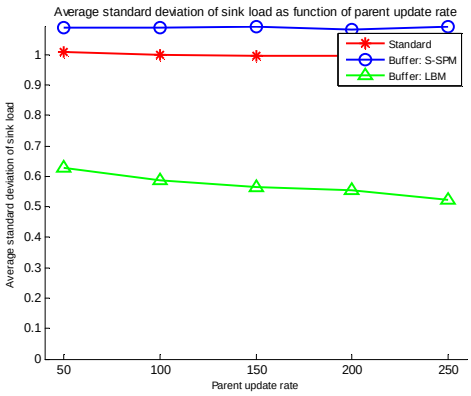


e) Network lifetime

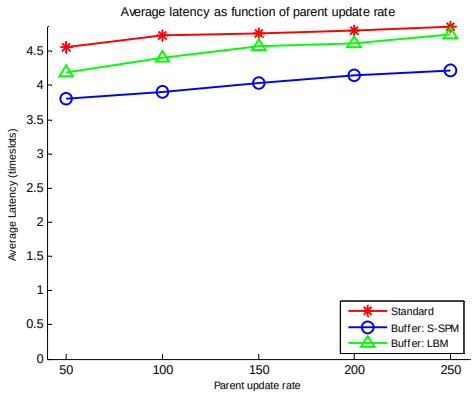


f) Energy efficiency

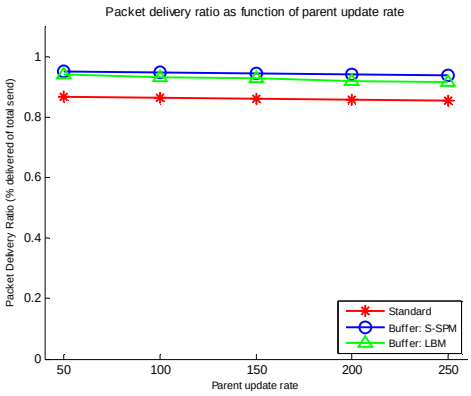
Figure 33 – Comparison of protocol performance as function of parent update rate in asymmetric clusters network



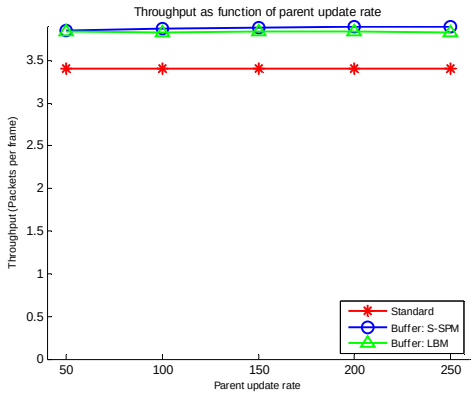
a) Standard deviation of load per sink



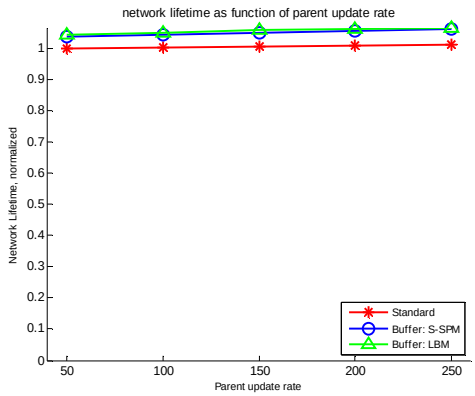
b) Latency



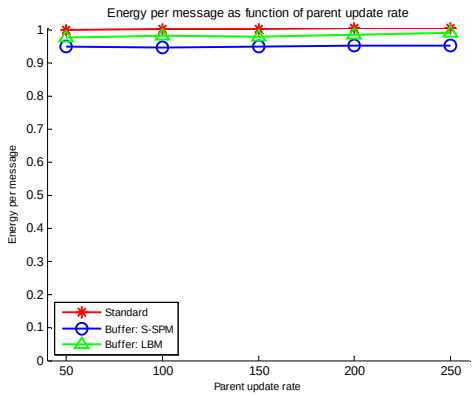
c) PDR



d) Throughput



e) Network lifetime



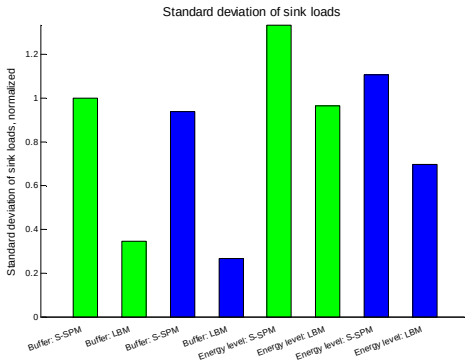
f) Energy efficiency

Figure 34 – Comparison of protocol performance as function of parent update rate in random network

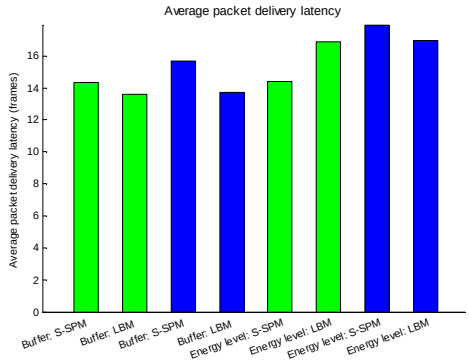
Effect of shortest path relaxation on routing metrics Buffer and Energy level

By using shortest path routing nodes forward data always to a neighbour closer to the sink. This is the easiest method and guarantees a loop free routing tree. However, bottlenecks in the tree – congested or nearly depleted nodes – cannot always be avoided

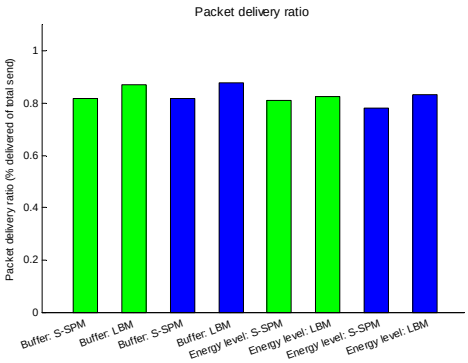
Simulations run on the asymmetric cluster network topology. Routing metrics Buffer and Energy level are taken, because they can benefit directly the best from the shortest path relaxation. Both metrics are simulated in S-SPM and LBM. Results without and with shortest path relaxation are coloured green and blue respectively. Standard deviation of sink load decreases a bit



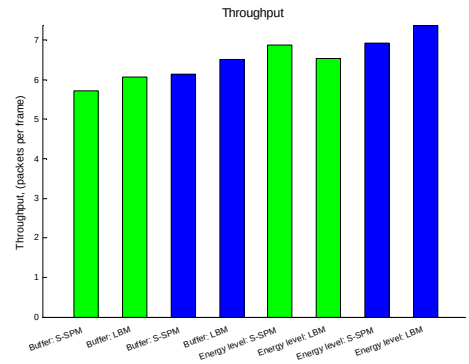
a) Standard deviation of load per sink



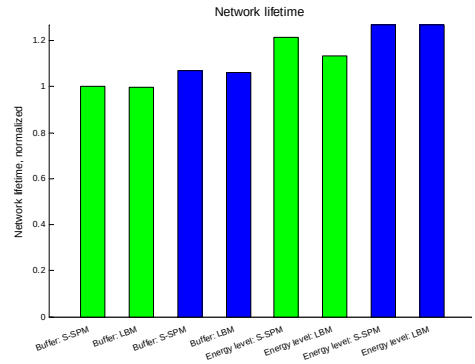
b) Latency



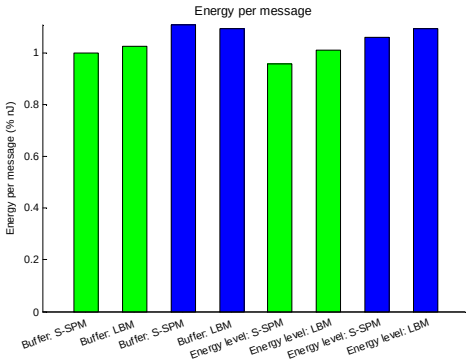
c) PDR



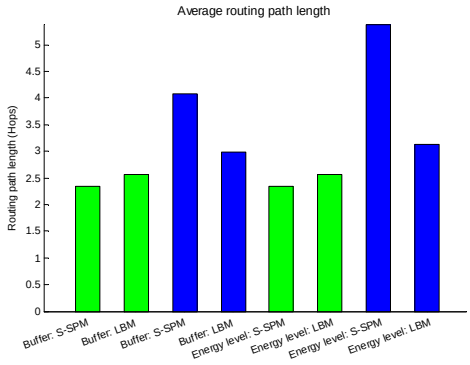
d) Throughput



e) Network lifetime



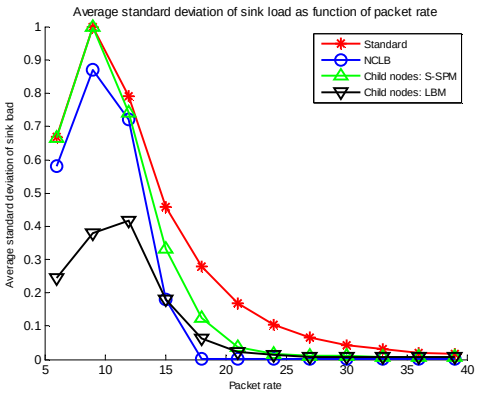
f) Energy efficiency



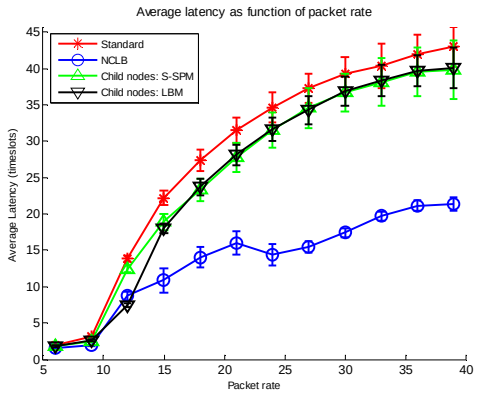
g) Routing path length

Figure 35 – Effect of shortest path relaxation on routing metrics *Buffer* and *Energy level*

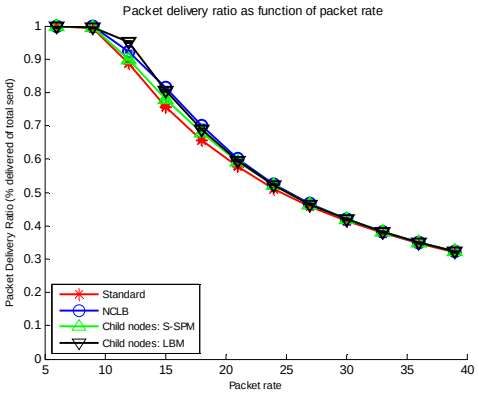
Comparison of protocol performance as function of packet rate



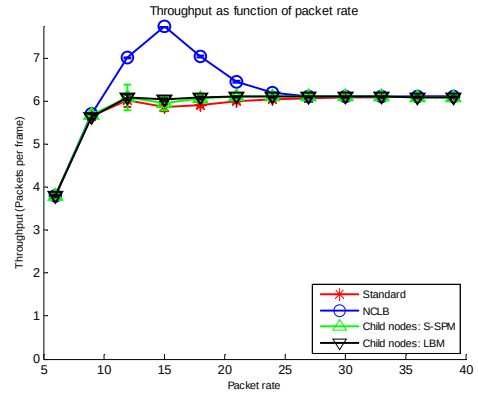
a) Standard deviation of load per sink



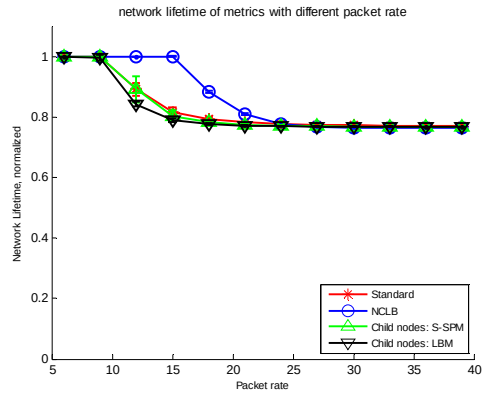
b) Latency



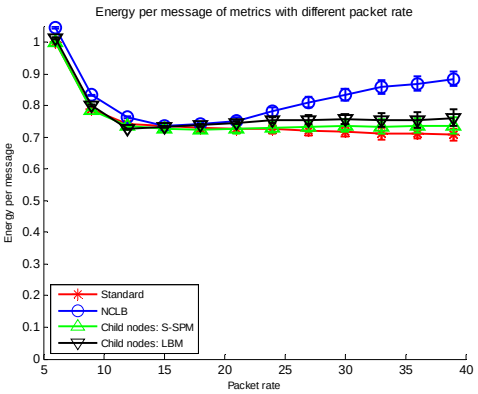
c) PDR



d) Throughput

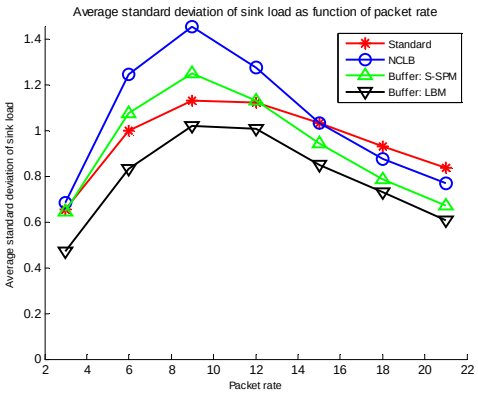


e) Network lifetime

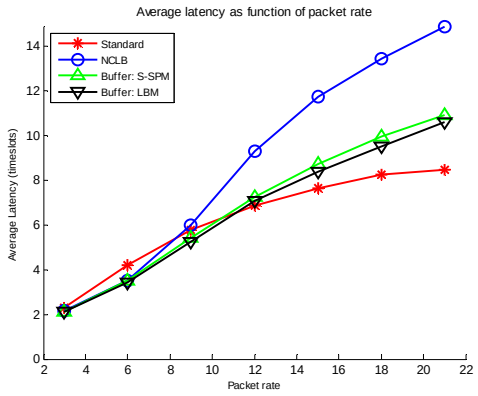


f) Energy efficiency

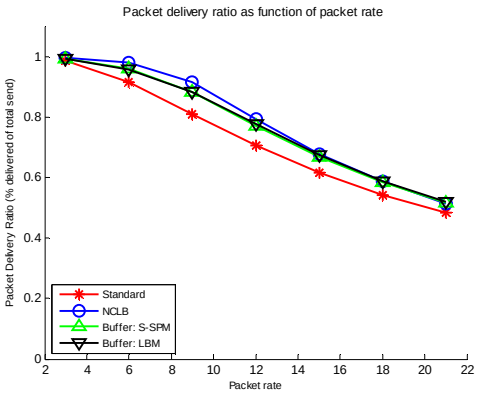
Figure 36 – Comparison of protocol performance as function of packet rate in asymmetric clusters network



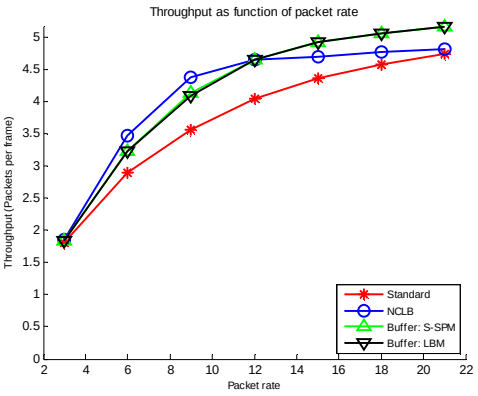
a) Standard deviation of load per sink



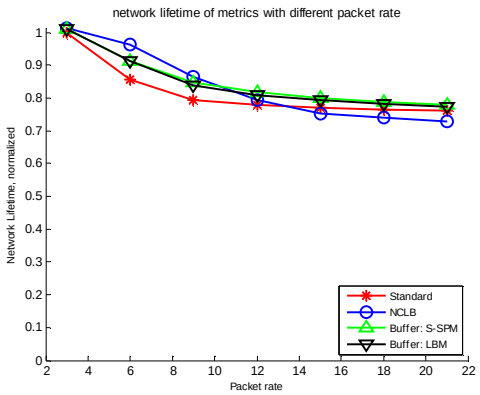
b) Latency



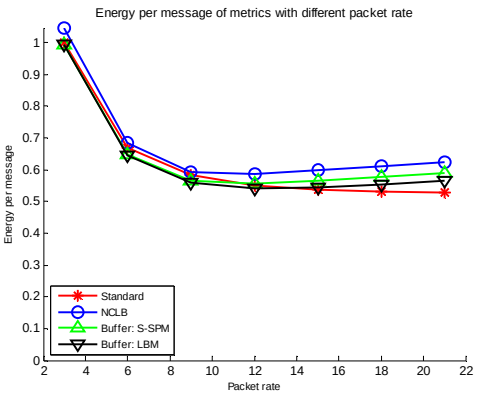
c) PDR



d) Throughput



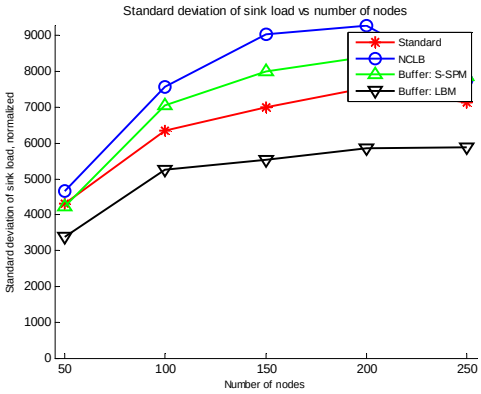
e) Network lifetime



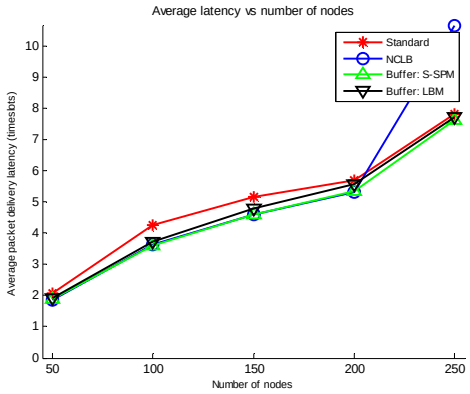
f) Energy efficiency

Figure 37 – Comparison of protocol performance as function of packet rate in random network

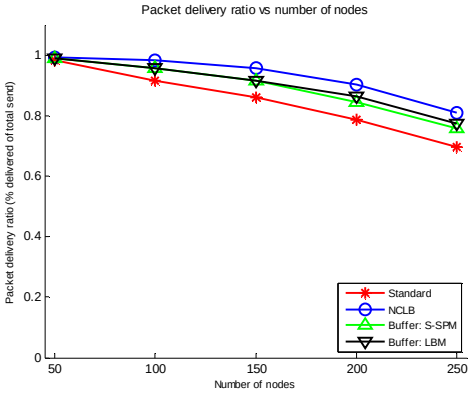
Scalability performance by varying the number of nodes and sinks



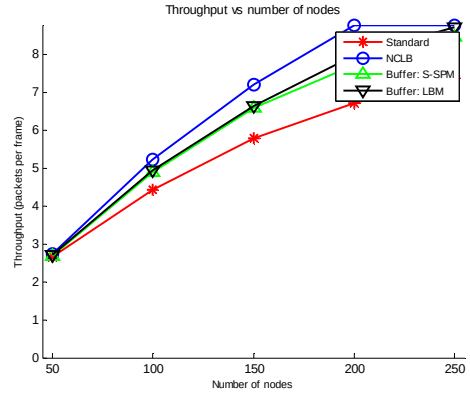
a) Standard deviation of load per sink



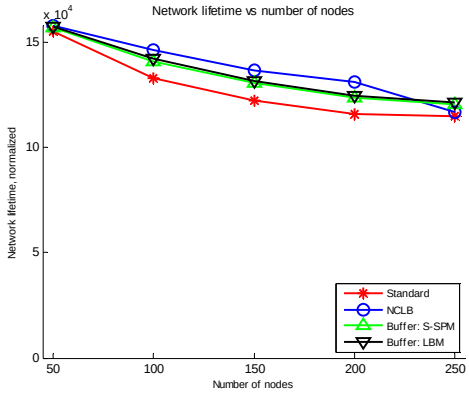
b) Latency



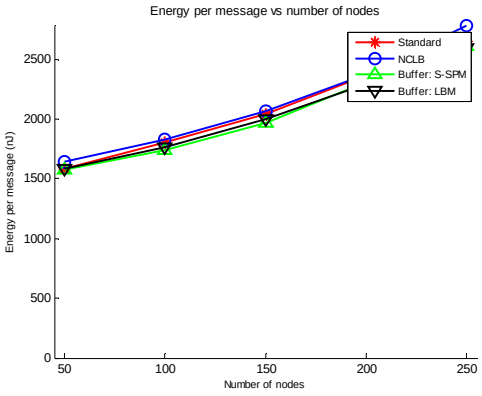
c) PDR



d) Throughput



e) Network lifetime



f) Energy efficiency

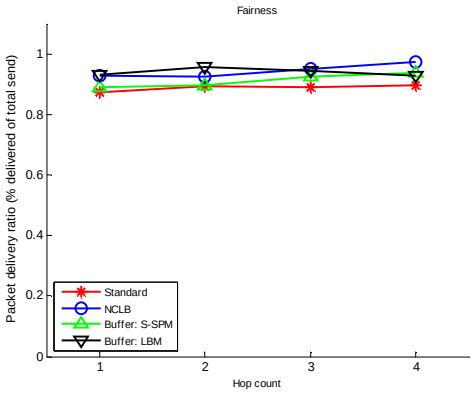
Figure 38 – Scalability performance by varying the number of nodes and sinks

Fairness of packet delivery ratio in three different network topologies

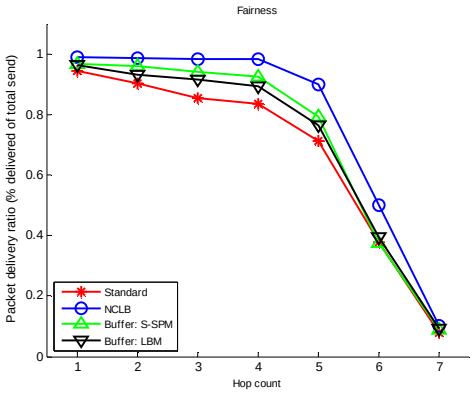
The *fairness* is a benchmark for the PDR as a function of the distance (measured in hops) from the node at which a packet is generated to the sink at which it delivered. Looking at the graphs of Figure 39 it is clear that the network type has a great influence on the fairness.

In the asymmetric clusters topology traffic load is more uniformly distributed over those neighbours, resulting in less packet loss. As a result, the PDR is more or less equal for all nodes of all distances.

In random formed networks are much more local bottlenecks, further away from the sinks. Therefore, congestion occurs further away from the sinks and thus the PDR is highest in nodes close around the sinks and lowest in nodes further away from those sinks.



a) Asymmetric clusters topology



b) Random topology

Figure 39 – Fairness of packet delivery ratio in three different network topologies