Karlsruhe Institute of Technology

Analysis of Scheduling and Topology-Control Algorithms for Wireless Ad Hoc Networks

Diploma Thesis of

Fabian Fuchs

At the faculty of Computer Science
Institute for Theoretical Informatics (ITI)

Reviewer: Prof. Dr. Dorothea Wagner
Advisor: Dipl.-Inform. Markus Völker

October 2011 – March 2012

Acknowledgement

I would like to thank Prof. Dr. Dorothea Wagner for giving me the possibility to compile this thesis, and my advisor Markus Völker for helpful discussions and his continual support. Also, I would like to thank everybody who supported me during the last six months.

The Lord has remembered us; he will bless us.

Psalm 115:12

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig angefertigt habe und nur die angegebenen Hilfsmittel und Quellen verwendet wurden.

Karlsruhe, den 30. März 2012

Ort, Datum (Fabian Fuchs)

Abstract

The ubiquity of wireless communication is one of the major innovations of the previous decades. In recent years especially wireless sensor networks evolved from theoretical consideration to practical application. In wireless sensor networks, energy conservation is crucial for the lifetime of the network. Since communication consumes most of the energy, research in recent years has focused on achieving more energy-efficient communication. One mechanism to improve the efficiency of communication is Time Division Multiple Access (TDMA) scheduling, which can be used to manage the medium access. TDMA schedules divide the time into time slots and assign those time slots to transmissions. In this thesis we study TDMA scheduling algorithms that enable efficient simultaneous transmission based on the Signal to Interference and Noise Ratio (SINR) model. In our simulations with the network simulator ns-3, we compare different SINR models and show that the throughput achieved with TDMA schedules is considerably higher than the throughput achieved with IEEE 802.11a CSMA/CA.

Another mechanism that has been considered in this thesis is topology control. Topology control aims at achieving more efficient communication by selecting communication links and thus reducing the energy required for transmission and minimizing interference. However, many well-known topology control algorithms have only been analyzed theoretically. We use simulations in ns-3 to study the throughput performance and the energy-efficiency of several well-known topology control algorithms such as the Yao Graph and XTC among others. For wireless communication according to the IEEE 802.11a standard, we observe that the throughput performance depends primarily on the number of hops that are on average necessary to transmit packets from one node in the network to another node, and only secondly on further aspects such as the signal strength of the used communication links (given they are above some threshold). Regarding energy efficiency the results of our simulations show that for fixed transmission powers the consumed energy is strongly correlated to the time needed to finish the transmissions, and to the average length of the selected communication links for variable transmission powers.

Contents

1. Introduction

The ubiquity of wireless networks in our every-day life is overwhelming. Wireless networks cannot only be used to conveniently access the internet but they also enable new areas of application. Today, many applications use sensors, often even a network of sensors. Due to the technical development, sensors can now be equipped with wireless communication instead of being connected by wire. A *wireless sensor node* is a micro-computer featuring sensing functionality in combination with a wireless communication device that may be used to connect with other wireless sensor nodes to a *wireless sensor network*.

The potential of wireless sensor networks opens interesting new fields of applications. One can, for example, equip each patient and each doctor in a hospital with a sensor node that senses information from the patients and transports them to the doctors or to a central station. Preferably, this node is small, relatively independent from infrastructure and can be used even if the patient is mobile. Another interesting application is crisis management. If the required infrastructure is destructed, a wireless sensor network can be used to communicate, sense critical areas or localize helpers. For an overview of additional applications of wireless sensor networks, we refer to [ASSC02, YMG08].

There are various constraints for the design of wireless sensor nodes, many of them depending on the application. However, there are some constraints that are shared by most applications. Nodes are usually not connected to the infrastructure and should therefore endure as long as possible without recharging. Also, the nodes should be small and low priced. Since small and cheap nodes usually are not equipped with a large battery, the used algorithms must ensure to conserve as much energy as possible. Since communication consumes a major part of the energy used by a sensor node, it is important to communicate efficiently.

Communication in wireless sensor networks has been a major field in research over the past years. In this thesis, we focus on scheduling and topology control algorithms. Scheduling algorithms compute a schedule that assigns each communication link a specified time in which the link is allowed to communicate. This avoids failures in communication due to interference and enables energy-efficient sleep and duty cycles. The aim of topology control is, to compute a subset of all possible communication links that allow communication such that energy is conserved and interference is minimized.

Sensor networks can be represented using a graph, which enables the application of graph-theoretic algorithms. In order to represent the sensor network as a graph, the sensor nodes can be modeled as vertices and possible communication between two sensor nodes can be represented by an edge between the corresponding vertices in the graph. Many topology control algorithms proposed by algorithm engineers use this representation of

sensor networks as a graph. Those topology control algorithms are often based on graph algorithms such as spanning trees or the Gabriel graph.

It is frequently assumed that interference can be minimized by using a sparse topology. However, this is not necessarily true, since limiting the set of communication links does not automatically avoid interference on neighbors. To examine how the sparseness as well as other properties such as the vertex degree or spanner properties influence the throughput of the different topologies, we examine some proposed topology control algorithms using the well-known network simulator ns-3 in this thesis.

1.1. Related Work

The aim of topology control is, to compute a subset of all possible communication links that allow communication such that energy is conserved and interference minimized. In the past years, research on topology control often considered topology control separated from other aspects like scheduling or routing. The models that were used are mainly graph-based and feature some well-known graph theoretic algorithms like the minimal spanning trees [LHS05, KPX07], the Gabriel graph or the Delaunay triangulation [GGH$^+$01]. Also some other algorithms have been proposed, among which the most popular ones are XTC [WZ03], Yao graph [Yao82] and cone-based-topology-control [WLBmW01], which is similar to the Yao graph.

It has often been assumed that the sparseness of a graph results in low interference without clear argumentation or proofs. In [BvRWZ04], Burkhart *et. al.* argument that interference is not effectively constrained by most topology control algorithms that were proposed. Afterwards, interference minimal topologies have been examined for different interference metrics in [MNL05, LZLD08, YDE11, LTWL11] and it has been shown that minimizing the maximum interference is NP-hard for the receiver interference model[1] [Buc08]. Very recently, interference and energy minimization have been considered jointly in [PSB12].

A more practical approach is the k-Neigh, which locally selects neighbors for each node such that the number of neighbors is equal to k [BLRS03].

Since retransmissions because of failures due to interference and listening on the wireless medium are energy-consuming, computing TDMA schedules became an important topic in research on wireless sensor networks. Spatial reuse TDMA, which allows more than one transmission to use the same time slot, additionally aims at minimizing the schedule length. First theoretical approaches to compute short schedules were mainly graph-based [GH01], and hence do not account for cumulated interference. Since the more realistic SINR model and the geometric Signal to Interference and Noise Ratio (SINR_G) model became popular in the theoretic research community, many schedules are computed along this interference measure [BBS06a, VKW09]. Unfortunately, scheduling is NP-hard in the general SINR model and both scheduling using common and variable but bounded transmission powers are NP-hard in the geometric SINR model [GOW07, VKW09].

As energy conservation is an important matter and both scheduling and topology control can improve energy conservation, these problems are also considered jointly in recent years. The first that joined the subjects were ElBatt and Ephremides [EE04] and others followed in recent years [BBS06a, VKW09]. Considering uniform transmission, [GWHW09], a first non-trivial approximation algorithms to compute a minimal schedules with an approximation factor of $\mathcal{O}(\log n)$ has been proposed. In [KV10], Kesselheim and Vöcking propose a distributed, randomized algorithm that computes an $\mathcal{O}(\log^2 n)$ schedule. Halldórson and Mitra improved this result to $\mathcal{O}(\log n)$ in [HM11]. Very recently, Kesselheim presented a constant factor approximation algorithm for the optimal selection of transmissions for one slot in [Kes11]. This yields an $\mathcal{O}(\log n)$ approximation for the scheduling problem.

[1] The receiver interference of a node is the number of transmission ranges it lies in.

A comparison between graph-based and interference-based TDMA schedules in [GH01] shows that interference-based scheduling can improve network capacity by up to one third for (temporarily) stationary situations. A more general outlook on protocol design beyond graph-based models, which leads towards the SINR model, finds similar improvements regarding the throughput of wireless networks [MWW06]. In [Mos06], Moscibroda *et al.* combine topology control with SINR based TDMA schedules.

An overview on algorithmic problems in wireless sensor networks can be found in [WW07]. For topology control we refer to [San05], while a general overview on wireless sensor networks can be found in [ASSC02, YMG08].

1.2. Contribution

Many existing approaches to the topology control problem have only been analyzed theoretically. It is often assumed that a low node degree minimizes the interference and thus yields energy-efficient communication. To the best of our knowledge, the performance of many of these topology control algorithms has not been analyzed and compared using a network simulator or a real network so far. In Chapter 5, we study the throughput performance as well as the energy efficiency of some topology control algorithms using the network simulator ns-3 to process traffic generated by random (possibly multi-hop) sender-receiver pairs.

Based on this simulation, we found that there is no direct connection between a low node degree and good performance regarding throughput or energy efficiency. In fact, topologies with a low node degree, like those based on the Euclidean Minimal Spanning Tree (EMST) or the XTC algorithms, usually achieve considerably less throughput than denser topologies. Regarding overall energy consumption for variable transmission powers we observed that the topologies based on the EMST, the XTC algorithm or an energy spanner achieve the best performance according to our measure.

However, both observations are not necessarily caused by the low node degree since it is due to the shorter edge length of those topologies that the transmission power could be reduced considerably. And it is mainly due to the increased number of hops that the throughput performance decreases.

Regarding absolute throughput performance, we found that different topologies fit best for different needs. For dense networks, usually a simple restriction on links up to a specific link strength or topologies such as the Yao graph or a 1.1-distance spanner maximizes the throughput performance while sparse topologies such as those based on XTC or a 1.1-energy spanner are more robust towards sparse networks and achieve the best performance for sparse networks in our comparisons.

Regarding energy consumption, the results are similar for fixed transmission powers as the length of the transmissions dominate the energy consumption, while for variable transmission powers those topologies that use mainly short links dominate as the transmission power can be reduced considerably. Namely these topologies are those based on the EMST, the XTC algorithm and the 1.1-energy spanner.

TDMA scheduling is considered an important mechanism to organize medium access as well as sleep cycles in wireless sensor networks. By applying TDMA schedules to the topologies considered, we get an additional criterion to analyze the performance of the topology control algorithms. As some wireless sensor network applications use TDMA instead of Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) to reduce energy consumption, this is an important metric for the topologies. We observed that regarding both the throughput performance as well as the energy consumption, the relative performance of the topologies has been similar to the relative performance in CSMA/CA. A restriction to communication links up to a certain signal strength yields the best throughput performance for both fixed and variable transmission powers as well as the best energy efficiency for fixed transmission power, while topologies that restrict on

very short links, such as the EMST, the 1.1-energy spanner or the XTC, are the most energy efficient for variable transmission powers.

1.3. Outline

This thesis is organized as follows. In Chapter 2 we introduce basic concepts as well as notations that are used throughout this thesis. Afterwards, in Chapter 3, we take a closer look at the network simulator ns-3, which is used for the simulations in this work and give an overview on the IEEE 802.11a standard for wireless communication. Chapter 4 considers the problem of TDMA schedules in wireless networks. We describe issues that exist regarding using TDMA schedules in ns-3 and propose a solution to these issues. We do also consider differences between TDMA schedules and IEEE 802.11a CSMA/CA and provide a simulation-based comparison between TDMA schedules and CSMA/CA regarding the throughput performance. In Chapter 5 we introduce several quality criteria of topology control as well as various topology control algorithms that are examined in this thesis. We discuss aspects such as the workload, the node density, variable transmission powers, and restrictions regarding the link strength based on simulations conducted with ns-3. We measure the performance of the topologies in terms of throughput and an expected overall energy consumption. The topologies are also studied in combination with TDMA schedules. Using only the communication links that are chosen by the topology control algorithm, routes for the random sender-receiver pairs are computed. For the links that are on those routes, a TDMA schedule is computed. The throughput and the energy consumption for the transmissions, which are processed according to the computed TDMA schedule, are considered. Finally, a brief conclusion and an outlook on future research directions are given in Chapter 6.

2. Preliminaries

In this chapter, we introduce some basic concepts and notations that are used throughout this thesis. In Section 2.1, we give some notations and definitions for graphs. Afterwards, an overview on wireless ad hoc networks and wireless sensor networks is given in Section 2.2.1 and in Section 2.2 respectively. Models that are used to represent wireless sensor networks such that a mathematical analysis is possible are described in Section 2.3.

2.1. Graphs

A *graph* is an ordered pair $G = (V, E)$ comprising a set V of *vertices* and a set $E \subset V \times V$ of *edges*.

For a graph $G = (V, E)$, G is said to be *directed* if the elements of E are ordered pairs, and is called *undirected* if such pairs are unordered. Within this thesis, an undirected graph is equivalent to a directed graph such that for every edge $e = (u, v) \in E$ there exists an edge $e' = (v, u) \in E$.

Each edge e may be assigned a *weight*, which is given by $\mathrm{w}(e)$. For simplicity we write $\mathrm{w}(u, v) := \mathrm{w}((u, v))$. In the following definitions, we assume that the weight is the Euclidean distance between the vertices, which is given by $d(u, v)$:

Definition 2.1. Given a graph $G = (V, E)$ and two nodes $u, v \in V$.

- A *path from u to v* (also called a *u-v-path*) is a sequence $(v_0, v_1, \ldots, v_p)$ of vertices such that $u = v_0$,$v = v_p$, and there exists an edge $(v_{i-1}, v_i) \in E$ for every $i \in \{1, \ldots, p\}$.
- A path from u to v is called *simple* if no vertices are repeated on the path.
- The *length* of a path $(v_0, v_1, \ldots, v_p)$ is defined as the sum over the weight of the edges on the path:
$$\mathrm{len}(v_0, v_1, \ldots, v_p) := \sum_{i=1}^{p} \mathrm{w}(v_{i-1}, v_i).$$
- A path $(v_0 = u, v_1, \ldots, v_p = v)$ from u to v is called *shortest path* if there is no path $(v'_0 = u, v'_1, \ldots, v'_p = v)$ from u to v with $\mathrm{len}(v'_0, v'_1, \ldots, v'_p) < \mathrm{len}(v_0, v_1, \ldots, v_{p'})$.
- The *distance* between two vertices u and v is defined as the length of the shortest path from u to v or. If no path from u to v exists in G, $\mathrm{dist}(u, v) := \infty$.
- A *cycle* is a path $(v_0, v_1, \ldots, v_r)$ with $v_0 = v_r$.
- If we require the path $(v_0, v_1, \ldots, v_{r-1})$ to be simple, we say $(v_0, v_1, \ldots, v_r)$ is a *simple cycle*.

Definition 2.2. For a graph $G = (V, E)$,

- G is connected, if for each pair $(u, v) \in V \times V$ a path from u to v exists.
- G is a *tree* if G is connected and G has no cycles.

Note that in a tree any two vertices are connected by exactly one shortest path.

2.2. Wireless Ad Hoc Networks

A Wireless Ad Hoc Network (WAHN) consists of so-called *nodes*: micro-computers that are able to communicate using a wireless network device. It is characteristic for WAHNs that nodes can communicate with each other without auxiliary devices such as routers. The nodes do not require previous individual setup, but once they are deployed in the environment, they are able to set up the wireless network ad hoc. The most typical features of WAHNs according to [San05, page 4] are:

- *Heterogeneous network:* The nodes in the network may be diverse. The only thing that the nodes must have in common is a wireless communication device, which enables them to communicate with other nodes in the network. It may be plain radio communication, acoustic signals or wireless communication according to transmission standards such as IEEE 802.11 or IEEE 802.15.4. Using, for example, wireless communication according to IEEE 802.11, devices such as smartphones, laptops, PDAs and others can form a wireless ad hoc network, since these devices usually are equipped with an appropriate communication device.
- *Mobility:* Usually most of the nodes are mobile, i.e., they move or can be moved.
- *Diffuse networks:* Wireless ad hoc networks are often scattered over a wide area. Multi-hop communication becomes necessary as the span of the network exceeds the transmission range. This is commonly assumed in applications and hence multi-hop communication must be realized.

A considerable amount of researchers have been attracted by WAHNs over the past few years. Nowadays the basic technology is available and there are algorithms for many problems. Still there are only few applications (for some, see [YMG08]). This is also due to the fact that although a lot of challenges have been tackled in the past few years, many of them are still not solved sufficiently. According to [San05, page 8], the main challenges are:

- *Energy conservation:* Due to mobility and the lack of infrastructure, nodes are usually battery equipped. Nodes should be handy and affordable, therefore the battery size is rather limited and the available amount of energy must be used as efficiently as possible.
- *Changing topology:* Nodes may move or power-down, hence communication partners may no longer be reached on the same route as before. Maintaining a correct and efficient topology in mobile networks is a complex task.
- *Low-quality communications:* In comparison with wired communication, wireless communication is error-prone. Since shadowing, fading, weather conditions and interference from other systems influence the communication, considering all factors requires sophisticated models.
- *Resource-constrained computation:* As mentioned, the nodes should be handy, affordable and energy efficient. This implies that the computational resources as well as network bandwidth are scarce.

- *Scalability:* For applications like vehicular networks or crisis-management networks, wireless ad hoc networks must span over large distances and the network may consist of hundreds or even thousands of nodes. Therefore, protocols and algorithms must scale efficiently up to very large networks.

The main aspects are that the nodes of a wireless ad hoc network must be cheap and as long-lasting as possible. Thus, minimizing the energy consumption is critical. However, energy is consumed in various ways:

- If the node is turned on, its components need energy to run. Sending components to sleep mode or disabling them is preferred.
- The more complex (and hence time-intensive) calculations are, the more energy they need. The CPU uses sleep or energy-saving modes to conserve energy if there are no calculations to be done.
- The communication device uses most energy in transmission mode. This may be up to two thirds of the total power needed by the node according to [San05, page 22]. Minimizing the number of transmissions as well as using sleep modes for the network device is vital for long lasting devices.

Depending on the application, one possibly can restrict to relatively low energy levels. But since there is a task that needs to be done, energy must be consumed. To choose and eventually tailor the algorithms used for this task is a key to minimize the energy consumption.

2.2.1. Wireless Sensor Networks

One of the main applications for wireless ad hoc networks are wireless sensor networks. Sensor networks do not have to be wireless. In fact, there are many applications for wired sensor networks, such as manufacturing machines, auto mobiles and security systems in buildings. As wireless technology became popular, applications for wireless sensor networks arose. The technology from wireless (ad hoc) networks has been merged with sensor functionality.

Most challenges of wireless ad hoc networks apply for wireless sensor networks, since the objectives regarding price, efficiency and persistence are similar. Depending on the actual application, the focus may shift to a subset of the objectives or other objectives and restrictions may be added. For some wireless sensor network applications, reliability and balance are essential: If the only sensor that detects a critical situation is powered-down due to an unbalanced workload, the whole sensor network may be useless. If, on the other hand, only an average temperature is to be measured, the network can easily cope with an outage or disconnection of smaller parts of the network.

In this thesis, we focus on algorithms for wireless sensor networks. Due to the similarities, our results are mostly applicable for both wireless sensor networks and wireless ad hoc networks. In Chapter 4 and Chapter 5, we study algorithms for scheduling and topology control, based on simulations using the network simulator ns-3 (see Chapter 3). The simulation results do not only tackle more theoretical questions, but they may also be used to chose the algorithms that fit best the application at hand.

2.3. Models for Wireless Sensor Networks

Due to the complexity of wireless communication concerning signal propagation and interference, researchers in algorithms for wireless sensor networks usually restrict to simplified models of the reality. This abstraction leads to mathematical models that enable a mathematical analysis of algorithms that are based on these models. However, due to the abstraction and simplification, it has to be ensured that good results for the models correlate to good results for real world applications.

In this section, we first describe models that are used to represent possible communication links between sensor nodes and afterwards introduce models concerning the interference of transmissions.

2.3.1. Communication Graphs

The topology of wireless networks can be represented with a graph: The nodes in the network correspond to the vertices in the graph, and connection links in the network correspond to edges between the corresponding vertices of the graph. We call this graph the *Communication Graph*. A weight can be associated to the edges, this may be the distance between the nodes, the energy used to communicate over this link, or the average signal strength achieved with this link.

Since signal propagation and the reception of a signal are non-trivial, there are various possibilities to model the correspondence between connection links in the network and edges in the graph. There are two models that are actively used to model wireless networks in the plane: *Unit Disk Graphs* and *Quasi-Unit Disk Graphs*. These models and models that are not restricted to the plane can also be found in [WW07].

Modeling a wireless ad hoc network using a graph is intuitive and allows to utilize knowledge from centuries of research undertaken within graph theory. There is a wide variety of algorithms for graphs available, hence numerous algorithms can be applied to wireless ad hoc networks. But since those algorithms assume a stable communication graph, an abstraction from the complexity of wireless signal propagation and reception is needed.

We consider propagation and path loss in more detail before describing models that decide whether an edge between two vertices should be added or not using the Unit Disk Graph and the Quasi Unit Disk Graph models.

2.3.2. Signal Propagation and Path loss

For both wired and wireless communication, the signal propagates along the used medium. Other than in wired communication using coaxial or fiber cable, where prediction of the strength and reach of a signal is easy, signal propagation for wireless communication depends on many factors: terrain, atmospheric conditions, weather conditions, and obstacles, among others.

Using the air as a medium, the main influence is the free-space loss of the signal as it propagates. At distance d to a sender sending with power P, a signal strength proportional to P/d^2 can be observed under ideal conditions. This is due to the fact that the energy of the signal distributes equally on the surface of a sphere that originates at the sender and grows with the speed of light.

As the energy density (or signal strength) decreases with the distance from the sender, three different ranges of the signal can be identified. The first one is the transmission range, within which the signal can be received with an error rate that can be compensated. Second, the sensing range, where the receiver can detect that the sender is sending but it is not able to decode the data. The last range is the interference range, where receivers can not detect that the sender is sending but other signals can be interfered by the sender's signal. We further introduce interference in Section 2.3.5.

The energy lost due to signal propagation and obstacles is often called *path loss*. Using a log-distance model, the path loss $\mathrm{L}^{\mathrm{dB}}(d)$ at distance d is modeled in dB[1]:

$$\mathrm{L}^{\mathrm{dB}}(d) = \mathrm{L}^{\mathrm{dB}}(d_0) + 10\alpha \log(d/d_0),$$

where α is the *attenuation coefficient*, which is usually assumed to be about 2 for free space propagation and between 3 and 5 for propagation in buildings. The path loss $\mathrm{L}^{\mathrm{dB}}(d_0)$ at

[1] In this thesis, we use dB for the ratio of two powers and dBm for absolute powers. An absolute power p in milliwatt equals $10 \log_{10}(p/1mW)$ dBm.

reference distance d_0 is a hardware dependent constants. The power $\mathrm{P^{dB}}_r(d)$ that is received from a receiver at distance d is (in dBm):

$$\mathrm{P^{dBm}}_r(d) = \mathrm{P^{dBm}}_t - \mathrm{L^{dB}}(d)$$

where $\mathrm{P^{dBm}}_t$ is the transmission power of the sending node in dBm. Alternatively, the power $\mathrm{P}_u(v)$ received at node v from sender u can be given in watt for $d \geq d_0$:

$$\mathrm{P}_u(v) = a \cdot \frac{\mathrm{P}_u(u)}{\mathrm{dist}(u,v)^\alpha} \tag{2.1}$$

where a is a hardware dependent constant and $\mathrm{P}_u(u)$ is defined as the transmission power of node u.

In reality, the path loss is not solely caused by the diffusion of the radio signals, but also by reflections on the ground and on obstacles, shadowing (e.g., by obstacles) of potential receivers, scattering and diffraction as well as so-called small scale fading.

Clearly, the signal propagation, is responsible for the general trend that the signal strength decreases with the distance. Shadowing, reflections, scattering and diffraction on the ground or on obstacles may cause worse received signal strengths for nodes that are, for example, in the shadow of objects that cause reflections.

Using a higher attenuation coefficient, this simple propagation model based on attenuation can be adapted to a lossier environment, e.g., one with more obstacles. The details regarding the path loss model are according to [WW07, page 28], while more on the fundamentals on wireless communication can be found in [Gar07].

2.3.3. Unit Disk Graph Model

In the *Unit Disk Graph* model, the transmission range is set to one fixed value. Communication between two nodes is assumed to be possible and successful if their distance is less than or equal to the transmission range.

Definition 2.3. (Unit Disk Graph, normalized)
Let G be a graph. G is a Unit Disk Graph, if $(u,v) \in E$ if and only if $\mathrm{dist}(u,v) \leq 1$.

Let the number of vertices in our graph be n. A geometric definition of Unit Disk Graphs is that each vertex is the center of one of n equal-sized circles with radius 1 and vertex u is connected to vertex v if and only if v is in the circle of u (and hence u is in the circle of v as well). The distances of the graph can be normalized, such that the transmission range equals 1.

The Unit Disk Graph models an idealized reality. The signal strength is sufficient as long as the receiver is in transmission range (or within distance 1, if normalized). Once this distance is exceeded, the signal strength abruptly decreases to a level that does neither enable sensing of the signal nor cause interference on other signals.

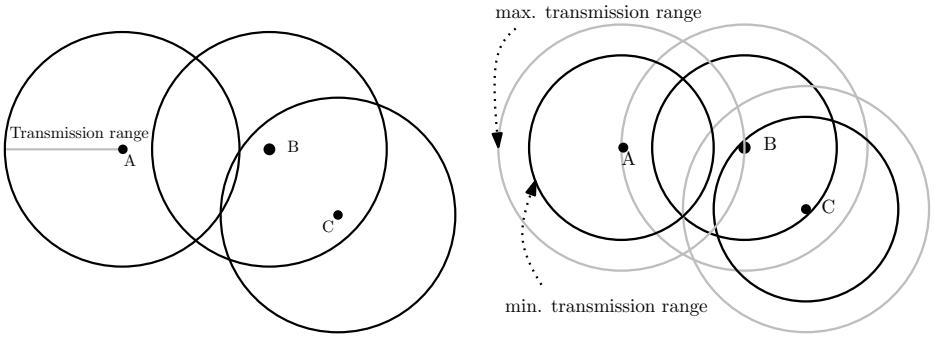
2.3.4. The Quasi Unit Disk Graph Model

Due to shadowing, reflections and scattering the transmission range is usually a lot more complex than a simple circle centred at the sender as in the Unit Disk Graph. To tackle this complexity, we need a more refined model. Considering the virtually infinite different environments, a reasonably complex but exact model can not be given.

The *Quasi Unit Disk Graph* is a graph that partially fills this gap: Again, a normalized graph with a maximum transmission range of 1 is used. For a parameter d with $0 \leq d \leq 1$, communication between two nodes with distance less than or equal to d is assumed to be possible and always successful in the d-Quasi Unit Disk Graph. Communication between two nodes with distance between d and 1 may be possible, while communication between nodes with distance greater than 1 is not possible.

Definition 2.4. (Quasi Unit Disk Graph, normalized)
Let d be a parameter with $0 \leq d \leq 1$ and $G = (V, E)$ be a graph. G is a d-Quasi Unit Disk Graph, if $\text{dist}(u, v) \leq d$ implies $(u, v) \in E$ and $\text{dist}(u, v) > 1$ implies $(u, v) \notin E$.

For the algorithmic models considered in this thesis, subgraphs of the Unit Disk Graph are considered. This is sufficient for our experiments, since—as in reality—successful transmission can not be guaranteed even for links that are in the Unit Disk Graph. An example of unit disks and quasi-unit disks can be seen in Figure 2.1.



(a) Unit disks (b) Quasi-unit disks

Figure 2.1.: For the unit disks on the left, the transmission range is depicted by the black circles. Node B can communicate with node C and vice versa, node A can not communicate with any other node. In the Unit Disk Graph, an edge would be added between B and C.
For the quasi-unit disks on the right, the communication between B and C is possible and communication betwen A and B may be possible. For nodes that are not in the grey circle of a node, communication between those nodes is not possible. In the Quasi-Unit Disk Graph, there would be an edge between B and C, there may be an edge between A and B, and an edge between A and C is not allowed.

2.3.5. Interference and the SINR Model

The models introduced in the previous section assume that communication is possible if the receiver is within a certain transmission range of the sender. But this is only one aspect of physical reception, since it does not only depend on the signal strength of an incoming packet but on the ratio of this signal's strength to the combined strength of other signals affecting the receiver to decide if reception of a packet is possible or not. Signals from other simultaneously sending nodes that are not desired at this receiver are called *interference*. Signals originating from the atmosphere and the electronic circuit at the receiver are called *noise*.

A relatively easy, graph-based approach to this problem is the construction of a *conflict graph*. In such a conflict graph G_{conflict}, for each communication link in the original graph a vertex e is added. An edge ec between two vertices e and c in the conflict graph G_{conflict} is added, if simultaneous transmission on the corresponding communication links is impossible. A set of transmissions is illustrated in Figure 2.2(a) and the corresponding conflict graph is given in Figure 2.2(b)

The conflict graph does account for noise and interference from individual transmission, but it fails to account for the summed interference of several simultaneous transmissions. Therefore, in the model interference is only a local phenomenon. In reality many interfering signals even far away add up and may prevent a transmission.

The so-called *Signal to Interference and Noise Ratio (SINR)* is not based on conflict graphs, but solely on the fading of the signals. At a receiving node r, the signal power

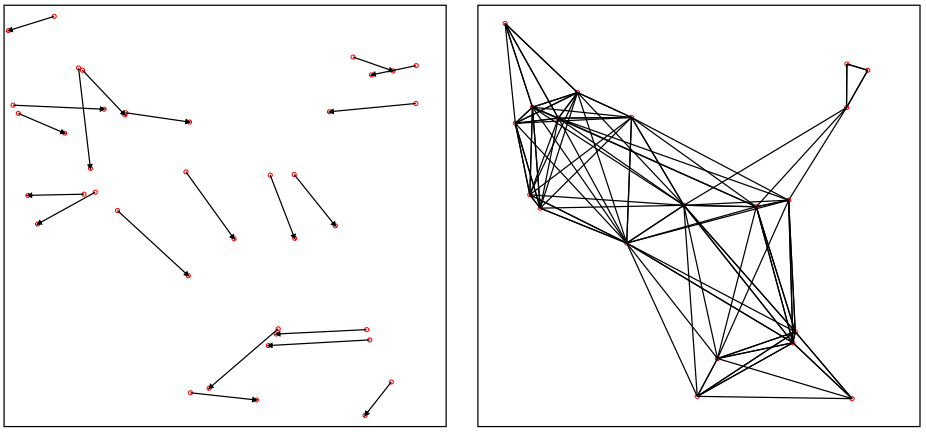


Figure 2.2.: On the left is a set of transmissions. On the right the corresponding conflict graph for a SINR threshold of $\beta = 10$. Each transmission pair in the left picture is displayed as a vertex on the right.

$\mathrm{P}_s(r)$ of the desired signal from node s is called the *signal*. The ratio of this signal divided by the noise N and the sum over the interference from all other nodes is the SINR as given in Equation (2.2).

$$\frac{\mathrm{P}_s(r)}{\mathrm{N} + \sum_{v \in V \setminus \{s\}} \mathrm{P}_v(r)} \geq \beta \tag{2.2}$$

The SINR model is also called the physical model and it is believed to resemble reality closely [WW07]. Due to the limitations of conflict graphs, research has lately focused on the SINR model.

It is not specified in the SINR model how the reception power $\mathrm{P}_s(r)$ is determined. In the *geometric* SINR model, $\mathrm{SINR_G}$, the power is calculated as a function of the distance:

$$\mathrm{P}^{\mathrm{g}}_s(r) := a \cdot \frac{\mathrm{P}_s(s)}{d(s,r)^\alpha}$$

with a hardware dependent constant a and the attenuation coefficient α depending on environmental factors. $d(s,r)$ is the Euclidean distance from s to r.

2.3.6. Dealing with Interference

We do now have a model that describes whether a reception is possible or not given the interference on the receiver during reception. Unfortunately, it is often unknown in advance, if and how much interference will occur, since it is unknown to the individual nodes when other nodes will start sending.

According to the OSI model[2], the Medium Access Control (MAC) layer handles access to the medium. In the MAC layer of wireless network devices, similar to wired Ethernet, usually Carrier Sense Multiple Access (CSMA) is used. Other than wired networks, wireless networks use CSMA/CA with *collision avoidance* instead of *collision detection*. A wireless sender is, due to its own signal, hardly capable of detecting another nodes transmission while it is transmitting a packet and hence collisions can not be detected. Therefore, it tries to avoid a collision, either by sending Request to Send (RTS)/Clear to Send (CTS) messages or by simply monitoring whether the medium is free for a randomly specified time before sending. If the signal of another node is received, after sending a RTS message or while monitoring the medium, the transmission is postponed [IEE07, page 256]. CSMA/CA is able to avoid most interference, but still there are two cases of sub-optimal

[2] A brief overview on the lower levels of the Open Systems Interconnection (OSI) model is given when the network simulator ns-3 is introduced in Section 3.1.3.

behavior that may occur. For both, a setup of 3 nodes is needed: A, B and C.

In the so-called hidden station scenario, A sends to B, while C is out of range of A but B is in range of C. C does not receive the signal of A's transmission and therefore may start a transmission. This however leads to interference at B, such that B may not be able to receive the correct signal from A.

The exposed station scenario is that A transmits data to B, and C receives the signal of A, but B would not receive C's signal if it would send. Here, C would not send even though it would not interfere with B receiving A's message.

For most use-cases, CSMA/CA is sufficient to avoid interference. In wireless sensor networks however, there may be a superior solution: Time Division Multiple Access (TDMA) schedules. TDMA manages the medium access by dividing the time in time slots and assigning those time slots to nodes that are allowed to send for the duration of the time slot. TDMA schedules require the nodes to send data only in time slots they are assigned to. This enables nodes to use the time slots they are not assigned to, to conserve energy by using sleep modes. Also, interference can be minimized, by computing time schedules such that the simultaneous transmissions keep interference on a low level. In Chapter 4, we consider TDMA schedules that compute schedules with low interference for wireless sensor networks.

3. Network Simulator ns-3

ns-3 is a time-discrete, event-driven network simulator, which is mainly developed for networking research purposes. ns-3's development initially began in 2006, while its first stable release was in June 2008. Since then, quarterly releases have been made. The current release is ns-3.13 [ns311a], which has been released on December 23, 2011.

In the following section, we give a brief historic background of ns-3 as well as an overview over the simulator in general. Afterwards, we briefly introduce the most important models for network modeling in ns-3 in Section 3.2. In Section 3.3, we introduce the IEEE 802.11 model implemented in ns-3 as well as a general setup for wireless networks. In Section 3.4, the routing algorithms for wireless networks that are implemented in ns-3 and used in Chapter 5 are described.

The ns-3 specific information in this chapter is based on the ns-3 manual [ns311b] and the documentation delivered with ns-3 itself (mostly the model library or the doxygen documentation) [ns311a].

3.1. Overview

ns-3 is partially based on several predecessors, using models and implementations from ns-2 [ns211], Georgia Tech Network Simulator (GTNetS) [gtn08] and Yet Another Network Simulator (YANS) [LH06]. As the name implies, it is designed to replace ns-2. Hence, in ns-3 several issues of the predecessor(s) were addressed:

Coding Style: At the time when ns-2 was developed, the Standard Template Library (STL) as well as other modern software engineering techniques have not been popular or available. ns-3 however uses state of the art software engineering as well as several design patterns, such as smart pointers, templates, callbacks and object aggregation.

Scripting Language: To avoid recompilations of the C++ code, and to provide a potentially easier scripting language to set up simulations, ns-2 is not solely written in C++ but makes heavy use of the scripting language OTcl. Nowadays, students are mostly unfamiliar to OTcl and it has been reportedly hard to debug the mixture of C++ and OTcl. Therefore, ns-3 uses pure C++ for the core and the models, while offering python bindings to set up simulations.

Code Contribution: Hundreds of models have been implemented in ns-2, but neither a coding standard nor consistent software testing or model verification have been enforced. This led ns-3 developers to abandon backward compatibility after careful consideration. Furthermore, a coding standard and code review for code contributions are enforced, and a test infrastructure has been established. This yields code contributions that are far more promising to be maintained even after the initial author(s) lost interest.

For a more detailed list and along with detailed motivations, we refer to [HLR08].

Over the years, many improvements have been made to ns-3 and additional models have been implemented (e.g., Internet Protocol Version 6 (IPv6) support, an Institute of Electrical and Electronics Engineers (IEEE) 802.16 Worldwide Interoperability for Microwave Access (WiMAX) module, etc.). Today, ns-3 is a well equipped network simulator that is maintained actively and implements a variety of different models. The most critical ones, like wireless models, have been tested and verified by the research community [PH09, PH10]. In most cases, those models are compatible and can be used in the main release. ns-2, in contrast, has some models that are still missing or under development for ns-3, though many of them are scattered among incompatible branches. One of the models, which is available in ns-2 but still under development for ns-3, is for example an implementation of IEEE 802.11.4, specifically IPv6 over low power WPAN (6LoWPAN).

Finally, ns-3 is developed as free software, licensed under the GNU GPL version 2 license. It is designed to run on Unix- and Linux-based systems. It is possible to run ns-3 on Windows using Cygwin. The recent stable release or the development trunk can be downloaded from the ns-3 homepage [ns311a].

3.1.1. Organization of ns-3

The organization of the ns-3 code can be divided into three parts. The first part is the core and commonly used parts of the simulator like packets and the event scheduler. The second part consists of the models that are used and needed for network simulation and the third part are auxiliary helper functions to simplify setting up a simulation environment and tests to ensure correct functionality throughout version updates.

Main parts of the core are defined by the modern object oriented approach of ns-3. We give more information on the main programming idioms, like callbacks and object aggregation, in the next section. ns-3, by default, features an event-driven simulator, which schedules the events according to its 64-bit internal simulation clock. A distributed simulator or a real-time simulator can be used instead, if the simulation is to be distributed on several machines or integrated into a testbed or virtual machine environment respectively.

Regarding the performance, maintenance and extensibility of a network simulator, packets are crucial. According to Section 16.1 in the ns-3 Model Library [ns311a], the design of the packets for ns-3 was focused on

(a) easy integration in real-world code and systems.

(b) fragmentation and concatenation should be supported.

(c) efficient memory management of packets.

(d) changes in the core of the simulator should not be necessary for the introduction of new packets, headers or tags.

The core of the simulator is the most stable part, as it is designed such that new simulation requirements should not imply changes in the core and the simulator.

The second part consists of some basic models and the models that are needed and hence contributed by the research community. This part holds models for computers, so-called nodes, from the network device down to the physical layer. It also holds applications and routing algorithms along with other models.

To simplify access to the variety of models, some commonly used models are equipped with so-called helpers. These helpers make it easy to connect objects. For example, a helper can be used to equip a container of nodes with network devices, or to install wireless communication devices including the MAC, physical and channel layer with corresponding parameters on several network devices at once. These helpers are defined for use in simulation scripts. Within the simulator itself, the use of helpers is not allowed.

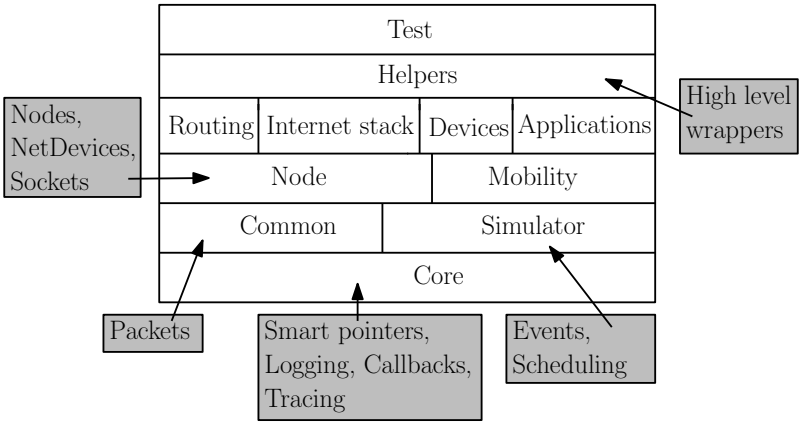


Figure 3.1.: Software organisation of ns-3, according to the ns-3 manual [ns311b].

Along with the coding standard, ns-3 introduced unit-test to verify the simulators behavior is as expected, after changes have been made. Unit-tests are enforced for new models, enhancement of models and bug-fixes.

An overview on the software organization of ns-3 is depicted in Figure 3.1.

3.1.2. Programming Idioms in ns-3

ns-3 uses a modern object-oriented approach as well as design patterns and programming idioms as mentioned previously. We now give a brief overview over some key parts of the simulator.

Base Classes: In ns-3, there are three base classes from which most classes in ns-3 inherit: The classes are `Object`, `ObjectBase` and `SimpleRefCount`. ns-3 classes must not inherit from those base classes, though, they offer some fundamental functionality. The `Object` base class implements the ns-3 type and attribute system, the object aggregation system as well as the smart-pointer system. `ObjectBase` implements the former two functionalities but does not implement the smart-pointer system. `SimpleRefCounter` does only implement smart pointers.

Object Aggregation: To address the various needs of different simulations, ns-3 makes it possible to aggregate (associate with) objects deriving from the `Object` base class to other objects implementing the same base class. If, for example, a node—using `Node`, which derives from `Object`—needs a position, the position can be aggregated to the node (using the `MobilityModel`), otherwise it does not have to be aggregated. This makes the node model lightweight and yet extensible to the different needs [WGG10, page 24].

TypeIds and the Config subsystem: Each class derived from `Object` may be given a unique `TypeId`, which is used to record meta-information for this class. The meta-information are mostly parameters for the object that can be easily accessed and changed using the TypeId in the so-called Config-subsystem. To set the default value of the maximum MAC-queue length in a wireless setup to 10, one simple call suffices:

```
Config::SetDefault("ns3::WifiMacQueue::MaxPacketNumber",
    UintegerValue (10));
```

Smart pointers: To avoid memory leaks and simplify memory management, ns-3 uses smart pointers to pass references. Smart pointers feature a reference counting system to keep track of the references of a counter and delete an object once the reference count falls to zero. For basic usage, smart pointers can be treated like a regular pointer (using the same syntax). To create objects, the function `CreateObject<ObjectName> ()` is used, where `ObjectName` must be a valid `TypeId`.

Callbacks: To increase modularity and reduce dependencies between objects, ns-3 makes extensive use of callbacks. Callbacks are C++ pointers that can be set to point at

a function which is specified at runtime. The use of callbacks enables ns-3 to dynamically connect objects such that the connection can easily be adapted to other simulation scenarios. The IP-implementation, for example, is enabled to connect to the layer above by offering a callback variable that must be set by the layer above. If the TCP protocol sets the callback to point at TCP, the IP layer is connected to the TCP protocol. However the IP-implementation could just as well be connected to a UDP protocol without adapting the IP-implementation - the UDP protocol simply has to make use of the callback designed to connect the IP-implementation with the layer above.

Tracing: Accessing classes and values of each object in the simulator is a complex task. One may want to observe only those packets dropped at the physical layer of a certain node. Therefore, the object `YansWifiPhy` implementing the physical layer of this node must first be identified using TypeIds and the Attribute System. Then, tracing sources defined at many objects to trace certain values can be used to retrieve the desired values. Here, we install a callback from the trace source PhyRxDrop to a self-defined method that increments our counter:

```
Config::ConnectWithoutContext("NodeList/[i]/DeviceList/[j]/$ns3::
    WifiNetDevice/Phy/$ns3::YansWifiPhy/PhyRxDrop", MakeCallback(
    &incrementDropCounter));
```

Ways to reach each object and trace sources are given in the ns-3 API documentation.

Random Variables: ns-3 uses the MRG32k3a random number generator by Pierre L'Ecuyer. There are several distributions implemented: `UniformVariable`, `NormalVariable`, `ErlangVariable` and many more. By default, ns-3 gives deterministic results. A randomization between different runs can be done by using a different seed or different run numbers. According to the manual, the "more statistically rigorous way to configure multiple independent replications is to use a fixed seed and to advance the run number" [ns311b].

As this gives only a brief introduction to the programming idioms of ns-3, more information can be found in the ns-3 manual [ns311b].

3.1.3. OSI Model in ns-3

So far, ns-3 implements protocols up to layer 4 of the OSI model. On top of layer 4, there are some applications that can be used to generate and receive traffic. Alternatively, own applications or traffic sources and sinks can be implemented. So far, layer 4 features the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP) only for Internet Protocol Version 4 (IPv4), though an expansion towards IPv6 is currently in its final stages of being finished.

For communication across the network, ns-3 offers different routing algorithms like global routing (which is for wired networks only), Optimized Link State Routing (OLSR), Ad-hoc On-demand Distance Vector (AODV), Destination-Sequenced Distance Vector (DSDV) or a static routing that may be used for user-defined routes. The routing algorithms are explained in more detail in Section 3.4.

Layer 2 consists of a higher and a lower MAC layer. The higher MAC layer handles active probing as well as a packet queue, packet fragmentation and packet retransmission if needed. The lower MAC layer is mainly responsible for data transmission to the physical layer and basic transactions like acknowledgement of received packets or RTS/CTS messages.

The physical and channel layer are responsible for the general properties of the used medium. Models exist for both wired or wireless communication. This layer includes models for propagation loss, propagation delay as well as general error models.

The physical-, channel- and MAC-layer for wireless communication are described in more detail in Section 3.3.

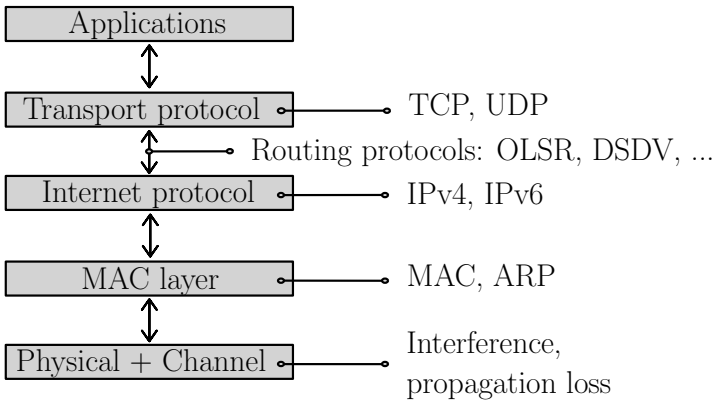


Figure 3.2.: The layers in ns-3 according to the OSI layer model.

3.2. Modeling Networks in ns-3

In order to simulate network scenarios, the simulator must provide models of the components of the network. In addition to the protocols of OSI layers 1-4, which are briefly described in Section 3.1.3, the following components are most relevant for modeling networks.

1. A *node* (`Node`) can be a network end system such as a personal computer, a network router or, as in our case, a wireless sensor node.
2. *Network devices* (`NetDevice`) enable the nodes to communicate. One node can host several network devices. Since every node needs a network device to communicate and via aggregation only one `NetDevice` could be associated to the node, network devices are organized in a list at each node. There are various network device models for the different communication modes, though devices for Ethernet and wireless communication via IEEE 802.11 are most common.
3. The *channel* (`Channel`) models the medium between the network devices. It might be a wired twisted pair medium, optical fiber, the air for wireless transmission or even water for underwater acoustic networks.
4. Data transmission is modeled via the transmission of network *packets* (`Packet`). Packets usually consist of one or more protocol headers and the actual data, called *payload*. In ns-3, packets are required to be exactly as they are in real networks.

The components given above mostly correspond to base classes of the network simulator. There is a wide variety of communication models that implement subclasses of the components given above. There are communication models for

- Long Term Evolution (LTE),
- Underwater Acoustic Network (UAN),
- WiMAX,
- Wireless LAN according to IEEE 802.11 a/b/g/n/e, and
- Ethernet.

Since the general setup is shared among the models, the more detailed view of the wireless model in Section 3.3 gives some reference on how other communication models may be set up. A detailed view on all communication models is given in the ns-3 manual [ns311b] and examples can be found along with the source code [ns311a].

3.3. Wireless Communication via IEEE 802.11

The wireless model for ns-3 is based on IEEE 802.11 and offers communication via 802.11a, 802.11b and 802.11g. The physical layer used for wireless communication is based on the implementation in YANS. The subclasses based on YANS are distinguishable by the "Yans" prefix. If there is an "Yans" implementation of a base class, we use the "Yans" implementation of the class, since Yans-Wifi is so far the only physical model for wireless communication included in ns-3 by default.

To understand the path each packet takes before, during and after transmission is important for understanding the network simulator as well as the protocols that are used for wireless communication. As it is necessary for our considerations to make and implement some modifications to the ns-3 code, is inevitable to—at least to some level—understand how the wireless transmission is defined in the IEEE 802.11a standard and how it is implemented in ns-3.

To give insight into how the IEEE 802.11 model implemented in ns-3 works, we first give a broad overview and then describe the stations on the path of a packet that is sent from one node to another node more detailed.

3.3.1. The IEEE 802.11 Model

The models used to implement the wireless communication according to IEEE 802.11, can roughly be assigned to four levels:

- MAC high models, which implement beacon generation, probing and association,
- MAC low models, which implement packet queueing, fragmentation and retransmisson as well as a Distributed Coordination Function (DCF),
- Rate control algorithms, and the
- Physical layer.

For easier understanding, we try to give the models roughly in the order they affect the path of a packet that is sent from one node to another node using the wireless communication via IEEE 802.11 in ns-3.

We begin with the `WifiNetDevice`, as this is the first model which is specific for wireless communication if a packet is sent down from higher layers. The `WifiNetDevice` holds together all objects related to the process of sending and receiving packets using wireless communication: `WifiChannel`, `WifiPhy`, `WifiMac` and `WifiRemoteStationManager`.

There are three implementations of the models of the higher MAC layer: The `ApWifiMac` implements the behavior of a wireless access point. It generates periodic beacons and accepts association requests. `StaWifiMac` is the MAC implementation for those stations that are not access points (i.e., regular clients). It implements active probing as well as re-association. For ad hoc networks, there is a so-called `AdhocWifiMac` which does not implement any beacon generation, probing or association. As the focus of this thesis is on sensor and ad hoc networks, we consider the `AdhocWifiMac` in the following.

The lower MAC levels consist of a `MacLow`, which implements functionality for RTS / CTS / ACK transactions. `DcfManager` and `DcfState` are used to coordinate access on the shared medium among the nodes. It relies on CSMA/CA, and optionally RTS/CTS functionality. `DcaTxop` handles packet queuing, fragmentation and retransmissions. It uses `DcfManager` to decide when a packet can be sent and `MacLow` to send the packet. To store the packet until the `DcfManager` allows the `MacLow` to send the packet, `WifiMacQueue` is used.

There are several rate control managers, implementing the base class `WifiRemoteStationManager`. Most managers use the SINR of the last packet(s) to decide at which bitrate the next packets should be sent. A manager with a constant bitrate and one with an idealized management is also available. As the differences between rate control managers are

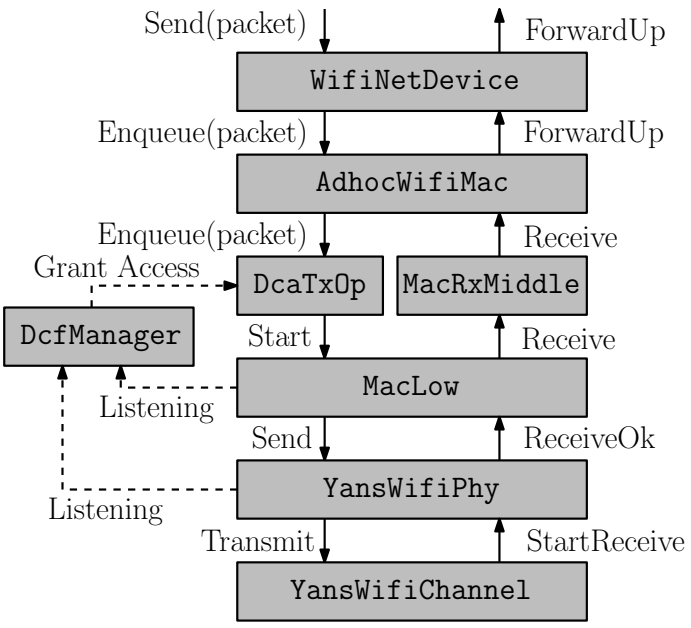


Figure 3.3.: Models of the link layer and physical layer as implemented in ns-3. Based on the ns-3 manual [ns311b].

not the focus of this thesis and the differences between some rate control managers and the idealized rate control manager are minor, we use `IdealWifiManager`.

If the MAC layer hands down a packet to the physical layer, it is sent to all other physical layers that are attached to the `YansWifiChannel`. Whether reception is possible or not is decided by the physical layer in `YansWifiPhy`. The channel offers access to various propagation loss and propagation delay models, which are subclasses of `PropagationLossModel` and `PropagationDelayModel` respectively.

The physical layer is modeled in `YansWifiPhy` and is fully described in [LH06]. There are mainly[1] three states in the physical model:

1. TX: A packet is currently transmitted using the physical layer on behalf of the associated MAC layer.

2. RX: A signal is received. The physical layer is waiting until the last bit of the packet is received to transfer it to the MAC.

3. IDLE: The physical layer is idle. It is not in the TX or RX states.

A packet is dropped, if its first bit is received while the physical layer of the receiver is in the TX or RX states, i.e., if it is receiving another packet or is sending one itself. If the receiver is in the IDLE state, the energy received in the first bit is compared to an energy detection threshold. If the energy is above the threshold, the physical layer moves to the RX state and waits for the last bit to be received. Once the last bit is received, the physical layer decides whether the packet can be successfully decoded or not. Therefore, the probability that the packet is received erroneous is calculated. This probability is called *packet error rate*[2] and is based on the SINR. More details on the reception process can be found either in [LH06] or the references given in the ns-3 manual [ns311b]

If the packet is received successfully, it is forwarded up to the `MacLow`, which sends a CTS if it has been a RTS packet or sends back an ACK to the sender and forwards

[1] There is also a state CCA_BUSY, which is equal to the IDLE state regarding packet reception and a SWITCHING state which indicates that the physical layer is switching the channel.

[2] The packet error rate considers changing SINR values during the reception of the packet.

the packet to `MacRxMiddle`, where duplicates are detected and fragments are recomposed. `AdhocWifiMac` and `WifiNetDevice` do simply forward the packet to the layers above.

The physical layer for wireless communication based on YANS is the only model that is delivered with ns-3 so far. But there exists a so-called PhySim-WiFi model, which models certain wireless communication more detailed. Yans-Wifi abstracts from the details of the packet transmission by considering only an average signal strength and the length of the packet. The Physim-Wifi model goes into the detail of the signal level to compute whether a packet can be received or not. However, the details of the PhySim-Wifi are coming with an excessively high running time (300 to 40000 times slower than the default model, depending on path loss and error models) [PMSH10]. Due to this we come to the conclusion that for our needs, the default Yans-Wifi is satisfactory, since the details of signal processing are not considered. For our simulations, an average signal strength for the whole packet is sufficiently detailed.

3.4. Routing Algorithms

In wireless sensor networks, communication between nodes that are not within transmission range may be necessary. Hence, the communication must use intermediate nodes that forward the packet to the receiver. The route that must be taken to reach a certain destination is computed by the routing algorithms. Since multi-hop communication is considered in Chapter 5, we describe routing algorithms that are implemented in ns-3 in this section. Since we consider wireless networks, OLSR, DSDV and AODV as well as a static routing algorithm are available. Using the static routing algorithm, we implemented hop-minimal and shortest-path routing algorithms that are described along with the algorithms directly implemented in ns-3 in the next sections.

3.4.1. Hop-Minimal and Shortest-Path Routing

Both hop-minimal and shortest-path routing are implemented using the ns-3 `Ipv4StaticRouting`. Both are very basic routing protocols that do not feature typical options of routing algorithms for (mobile) ad hoc networks such as flooding or the evasion of broken links.

Since the static routing algorithm in ns-3 does not offer predefined routes in the wireless environment, the possible routes must be computed and added to the routing protocol. To compute the routes, a simple breadth-first search finds the hop-minimal or the shortest path to each node respectively. Starting from a source s, for each processed node the algorithm stores the hop-minimal or shortest node on the (either hop-minimal or shortest) path from the source that led to its processing. This node is used as gateway in the routing algorithm from the source to the processed node.

3.4.2. Optimized Link State Routing (OLSR)

The OLSR routing protocol is specifically designed for mobile ad hoc networks. It is a proactive routing protocol and hence establishes routes before they are demanded. To find its one and two hop neighbors, each node uses OLSR-*hello* messages. Additionally, each node selects special nodes, so-called multipoint relays such that all one- and two-hop neighbors can be reached through them. Each multipoint relay node does then distribute its neighbor information to the nodes that selected it as a multipoint relay. In contrast to the classical approach, where each node retransmits each message when it is received the first time, this reduces message overhead.

According to RFC 3626 *Optimized Link State Routing Protocol (OLSR)* [CJ03], OLSR provides hop-optimal routes and particularly suits large and dense networks. The ns-3 implementation of OLSR has been developed for ns-2 and was ported to ns-3. It is mostly compliant with RFC 3626, except for MAC layer feedback, which is missing in the ns-3 implementation of OLSR.

3.4.3. Destination-Sequenced Distance Vector (DSDV)

DSDV is based on the Bellmann-Ford algorithm [CLRS09, pages 651-655]. DSDV uses a routing table to keep track of routes to possible destinations and regularly exchanges these tables with neighbors. One problem that arises in the Bellmann-Ford-based routing algorithms is the problem of induced routing loops. Considering 3 nodes, A, B and C where A can only communicate with B, B can communicate with A and C, and C can only communicate with B. If A breaks down, B notices that it can not reach A anymore as A does not respond. However, B does also receive updates from C which tells B that A is just 2 hops away from C. This is not anymore true and induces a (temporary) routing loop which lasts (in this case) for another two updates. One of the major achievements of DSDV is that it avoids this routing loop problem by using incrementing sequence numbers generated by the destination node. To update the routes, full or incremental updates are distributed between nodes. For highly dynamic, long-lasting networks, the gradually incrementing sequence number may become an issue.

In the ns-3 implementation of DSDV, a node sends out updates on its routing table whenever the routing table is changed.

3.4.4. Ad-hoc On-demand Distance Vector (AODV)

In contrast to OLSR and DSDV, AODV is a reactive routing protocol, and therefore discovers routes once they are required. For each neighbor of a node, a list that holds destination IPs that are likely to use this neighbor as a next hop towards the destination is hold. Then, to find a new route, the source node sends out a route request that is forwarded until the target, or a node with a route to the target, is found. Discovered routes are stored, but invalidated after a specified time. To avoid the Bellmann-Ford loop problem, AODV also uses monotonously increasing sequence numbers, which may become an issue on highly dynamic, long-lasting networks.

The ns-3 implementation of AODV is based on RFC 3561 *AODV Routing* [PBRD03]. Some issues that concern cooperation of OSI layers are claimed to be not described in the RFC. The ns-3 implementation of AODV hence implements heuristics to (1) detect and avoid unidirectional links (2) use *hello* messages to detect broken links and (3) detect duplicate packets. For wireless transmission, the use of *hello* messages is critical, since those messages are transmitted using a lower bit rate than usual packets. Therefore, they travel further and are more resistant to interference.

The performance of AODV-based multi-hop transmission has been relatively poor in our initial experiments. Problems with the implementation of AODV in ns-3 as well as a similarly performance have also been reported in [NCc+11]. Therefore, we do not consider the AODV routing algorithm in our experiments in Chapter 5.

4. Scheduling

In this chapter we consider scheduling algorithms that compute Time Division Multiple Access (TDMA) schedules for transmissions such that the transmissions can be processed simultaneously without failure due to interference.

We will first give an introduction to SINR-based scheduling in the next section. Then, some issues regarding the simulation of TDMA scheduling in the network simulator ns-3 are discussed in Section 4.2. An overview on the considered scheduling algorithms is given in Section 4.3. In Section 4.4, we describe the experimental setup that is used to conduct the experiments. The experiments themselves are described and their results are presented in Section 4.5. This chapter is concluded with an overview over the results of this chapter in Section 4.6.

4.1. Introduction

TDMA schedules manage medium access by dividing the time into time slots and assigning these time slots to wireless sensor nodes, which are only allowed to send during the assigned time slots. In the assigned time slots, a node may access the medium for transmission or reception. In time slots the node is not participating, it can use its sleep modes to conserve energy. Since energy is a very valuable resource for wireless sensor nodes, the use of TDMA scheduling is tempting.

4.1.1. SINR-based TDMA Schedules

Before going into the details of TDMA scheduling, we will first introduce some notations: A *transmission pair* t consists of a sender s and receiver r, $t := (s, r)$, and is associated with a specified amount of data that must be transferred. For now, we assume the amount of data to equal the amount of data transferable in one time slot. To compute a schedule, each transmission pair must be assigned one time slot in order to transmit the associated data.

To achieve schedules such that the interference between the transmission pairs in each slot is low enough to enable a successful transmission, usually the Signal to Interference and Noise Ratio (SINR) model is used. Using the SINR model, we can decide whether a set of transmission pairs can be active in the same time slot (i.e., whether they can transmit their data simultaneously). The SINR model is introduced in Section 2.3.5. We assume the SINR threshold β to be given as $\beta = 10\,\mathrm{dB}$ in this chapter.

Assume a set S of transmission pairs, such that all senders can transmit their data simultaneously (i.e., in one time slot) according to the SINR model. Then, we can decide based on the SINR formula whether a transmission pair $t := (s, r)$ can be processed while

the other pairs in S are transmitting. If it holds that

$$\frac{\mathrm{P}_s(r)}{\mathrm{N} + \sum_{(s',r') \in S} \mathrm{P}_{s'}(r)} \geq \beta, \tag{4.1}$$

r can receive the signal of s with a signal strength that is sufficiently high to decode the signal, even though the other pairs in S are sending. However, this is not sufficient, since it is not clear how t influences the transmissions that are yet in S. Before adding t to S, we have to check for each $t_j \in S$ whether it can still transmit its data successful according to the SINR model while the transmission pairs in $S \cup \{t\}$ are transmitting. The notation used in this formula has been introduced in Section 2.3.5. In short: $\mathrm{P}_s(r)$ is the energy received at node r when s is sending, $\mathrm{P}_s(s)$ is defined as the sending power of node s, and N is the background noise induced by the circuit and the environment.

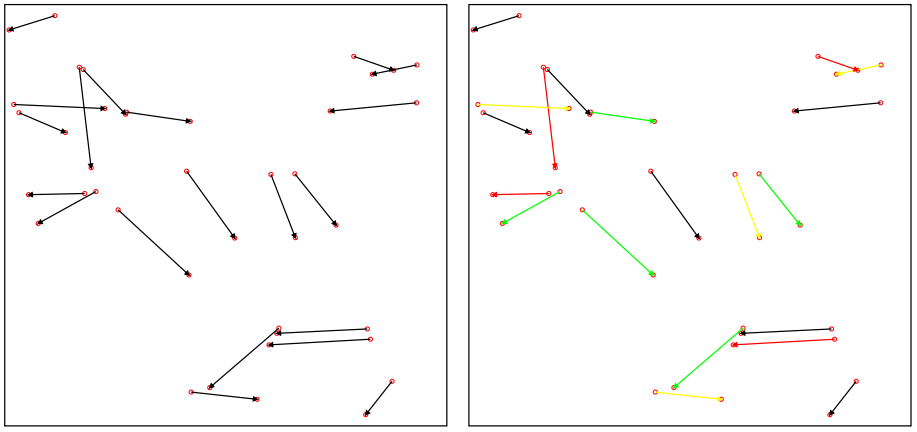


Figure 4.1.: Input and output of the TDMA scheduling problem. On the left is a set of transmissions, i.e. sender-receiver pairs. On the right, a possible assignment fo these transmissions to TDMA time slots is depicted. Each color corresponds to one time slot.

Sending wireless signals with higher energy increases the energy with which the signal is received at the receiver. Hence, it enables the sender to reach receivers further away. At the same time, however, the interference on receivers that are not interested in the signal is increased accordingly. As we want to minimize interference and ultimately conserve energy, sending with lower power, limits the transmission range while also limiting the interference on other nodes. Therefore, the lowest possible sending power must be chosen such that the intended receiver is able to decode the signal according to the SINR equation.

Let T be the set of all transmission pairs. We can now define the TDMA scheduling problem, which is the problem of finding a slot assignment for each transmission in T such that for each transmission in the time slot the SINR equation holds and the total length of the schedule is minimal. We call an assignment of transmissions to a time slot such that the SINR equation holds for each transmission in the corresponding time slot *valid*. The definition of the scheduling problem for a common transmission power of all nodes is given in Definition 4.1.

Definition 4.1. (TDMA scheduling problem with common transmission power)
Given a set of transmissions T with common transmission power P and time slots $S_1, S_2, \ldots$, find an assignment of the transmissions in T to time slots S_i such that

(a) For each S_i: for each $t = (s, r) \in S_i$ the SINR equation holds, i.e.,

$$\frac{\mathrm{P}}{\mathrm{N} + \sum_{(s',r') \in S_i \setminus \{t\}} \mathrm{P}_{s'}(r)} \geq \beta$$

(b) $S_1 \cup \cdots \cup S_k = T$ for a minimal value of k.

For variable transmission powers, the transmission power of each pair can be adjusted. This may lead to shorter and hence more efficient schedules. The drawback is that for each pair that is added to an existing time slot, the transmission powers of the pairs in the slot may need to be updated. The definition of the problem of scheduling with variable transmission power is given in Definition 4.2

Definition 4.2. (TDMA scheduling problem with variable transmission power)
Given a set of transmissions T, a maximum transmission power P_{max} and time slots $S_1, S_2 \ldots$, find transmission powers and an assignment of the transmissions in T to time slots S_i such that

(a) For the transmission power of each transmission it holds that $\mathrm{P}_s(s) \in [0, \mathrm{P}_{max}]$

(a) For each S_i: for each $t = (s, r) \in S_i$ the SINR equation holds, i.e.,

$$\frac{\mathrm{P}_s(r)}{\mathrm{N} + \sum_{(s',r') \in S_i \setminus \{t\}} \mathrm{P}_{s'}(r)} \geq \beta$$

(b) $S_1 \cup \cdots \cup S_k = T$ for a minimal value of k.

The TDMA scheduling problems as given above are defined for the general SINR model. In Section 2.3.5, we described $\mathrm{SINR_G}$, a geometric SINR model that extends the SINR model with a defined path loss depending on the distance between the communication partners. The scheduling problems can be easily transferred to the $\mathrm{SINR_G}$ model by defining the powers according to the geometric model.

Assuming the $\mathrm{SINR_G}$ model, the following results have been shown: The scheduling problem using uniform transmission powers is NP-hard [GOW07]. The scheduling problem using variable but bounded transmission powers is NP-hard [VKW09]. As the scheduling problem is NP-hard for the geometric model, scheduling is also NP-hard for the general SINR model [GOW07].

4.2. Issues regarding a TDMA Simulation in ns-3

This section gives a short overview on issues related to transmitting data according to TDMA schedules in the network simulator ns-3.

A main problem that has come to our attention is that scheduling algorithms according to the SINR model usually do only account for signals that are emitted by the senders, and interference that occurred at the receivers. For common wireless networks such as the IEEE 802.11, signals are emitted by both sending and receiving nodes. While the receiver is interfered by undesired signals, the sender may be blocked due to the medium access mechanism CSMA/CA, triggered by undesired signals.

Before considering this problem, we will first discuss how scheduling can be integrated in the IEEE 802.11 wireless models used by ns-3 in the next section. Section 4.2.2 discusses how common MAC functionality such as ACKs influence the performance of SINR-based TDMA schedules. In Section 4.2.3, finally, we consider how we can solve the mentioned problem and enable the use of TDMA scheduling using the IEEE 802.11 models in ns-3.

4.2.1. Integrating TDMA Schedules in ns-3

The wireless communication offered by ns-3 that is suitable for wireless sensor and ad hoc networks is so far restricted to the IEEE 802.11 models described in Section 3.3.1. As the IEEE 802.11 models implement medium access using the CSMA/CA mechanism as well as the Request to Send (RTS)/Clear to Send (CTS) handshake if packets are larger than a threshold, TDMA schedules can not be simulated without modification of the wireless model.

There is a wide consent in the research community that scheduling is usually done in the MAC layer, since it handles access to the medium [ASSC02, YMG08]. In ns-3, however, there is no MAC layer available that enables TDMA scheduling for wireless communication. This due to the fact that medium access using TDMA is not considered in the IEEE 802.11 standard.

Since the MAC layer of ns-3 is very detailed (cf. Section 3.3.1) and hence complex, an adaptation of this model without risking undesired behavior was not considered possible. We therefore implemented scheduling using the application layer and adapt the MAC layer to the needs of TDMA scheduling as good as possible.

4.2.2. Link Layer Acknowledgements

Many common MAC protocols rely on bi-directional communication, due to MAC acknowledgements (ACKs) and possible RTS/CTS handshakes. RTS/CTS handshakes can easily be avoided in the wireless model of ns-3 by using packets smaller than 2200 bytes. Since we usually use packets with a payload of 512 bytes, this is not a problem.

The bi-directional communication introduced by link level ACKs however is seldom considered in scheduling algorithms. This may be partially due to the fact that common 802.11 wireless MAC protocols do not yet offer TDMA scheduling functionality. Another reason may be the emergence of wireless communication according the IEEE 802.15.4 standard for "Wireless Personal Area Networks", namely ZigBee [Erg04] and 6LoWPAN [MKHC07], which are designed for very low bandwidths and low energy consumption. IEEE 802.15.4 does offer both, a slot-based scheduling as well as the ability to disable acknowledgements.

However, for applications that have high demands on the bandwidth, using image or video processing for example, such a low bandwidth does not suffice and wireless communication using 802.11 or similar standards is required. This requirement is known and various MAC protocols have been proposed that fit the needs of wireless sensor and ad hoc networks. However, many do still use ACKs [vDL03, SKS06], because without using ACKs the sender can not be sure whether the receiver received the message sent to him or not. Also, ACKs enable the sender to adapt its sending concept according to the ACKs received.

Even though ACKs are rather short, they interfere with other packets on a regular basis. Therefore, we need a more general inspection of the SINR constraint for each transmission that should be added to a time slot. It must be considered that sending nodes are interfered while waiting for a free channel if they receive packets, as well as that receiving nodes are interfering other nodes by sending ACK messages. We assume that for each transmission, either the sender or the receiver is sending at a time.

To account for those changes, we can use the following rule. If a transmission pair t should be added to an existing set of simultaneous transmissions S, Equation (4.2) and Equation (4.3) must hold for every pair $(s, r) \in S_t := S \cup \{t\}$.

$$\frac{\mathrm{P}_s(r)}{\mathrm{N} + \sum_{(s',r') \in S_t \setminus \{(s,r)\}} \max(\mathrm{P}_{s'}(r), \mathrm{P}_{r'}(r))} \geq \beta. \tag{4.2}$$

$$\frac{\mathrm{P}_r(s)}{\mathrm{N} + \sum_{(s',r') \in S_t \setminus \{(s,r)\}} \max(\mathrm{P}_{s'}(s), \mathrm{P}_{r'}(s))} \geq \beta. \tag{4.3}$$

Note that by the constraints we added, we ensure that the SINR equation holds for both, receiving and sending nodes. This model has been proposed in [BBS06b]. We call this model the bi-directional SINR model and the previously introduced model the general SINR model. This approach can be used for both common or variable transmission power scheduling.

4.2.3. Simulating TDMA using IEEE 802.11 CSMA/CA

As discussed earlier, the IEEE 802.11 models do not implement TDMA but are restricted to CSMA/CA. Since we do not consider changing large parts of the ns-3 simulator as an option, we have to adapt the models such that they can be used for TDMA schedules without actually changing their code.

The main difficulty is, to force the sender to ignore the CSMA/CA mechanism and thus to force it to start its transmissions even though it may be reached by packets from other nodes sending simultaneously. For each packet that reaches a node, the energy of this packet is compared to the energy detection threshold. If the energy is higher than the threshold, the node tries to receive the packet and notificates the CSMA/CA mechanism that the medium is used. To avoid these notifications, the energy detection threshold of the sending nodes can be increased. However, it can not be set to an arbitrary high level, since ACKs from the receiving node must be received.

Using the bi-directional SINR model, it holds that the energy of all other nodes that are sending in the same time slot (both sending and receiving nodes) is not larger than the signal of the intended partner in the transmission pair. In fact, all interfering signals received at a node are exceeded by the intended signal by a factor of at least β.

Using this knowledge, we set the energy detection threshold such that the signal of the receiver can be received, but all other incoming signals are not considered. Note that this is not according to any standard. It is a workaround to disable CSMA/CA in ns-3. Since messages received at the sender are also not considered by scheduling algorithms, the workaround is acceptable.

Using the general SINR criterion for TDMA schedules, however, there seems to be no way to easily bypass CSMA/CA in ns-3. This is because the interference that affects the sending node is not strictly limited. Since the distance of the sender to another sender may be arbitrarily small, the signal strength from the interfering sender may be considerably higher at the sender than at the receiver.

Another mechanism that hinders TDMA schedules is the RTS/CTS handshake mechanism. As mentioned before, RTS/CTS is only used for large packets in ns-3 and hence this handshake mechanism can be circumvented by using packets with 512 bytes of payload, which is the default size of the used application.

To force the MAC layer to send packets only during the assigned time slot, we need to make sure the MAC layer runs out of packets once the upper layer that implements TDMA stops sending packets (i.e., the time slot is over). Since modifications on the MAC layer have not been an option, all queues from the application layer to the physical layer had to be disabled. Since those queues are needed, a small number of packets has been allowed. For an exact description of the used parameters and modifications, we refer to Section 4.4.3.

4.3. Algorithms

In this section we introduce the—rather intuitive—algorithms considered for the scheduling experiments. Due to timely limitations of this thesis we refrained from an extension on more elaborate scheduling algorithms.

4.3.1. GreedySINR

After ordering the transmissions according to their link gain (which equals an ordering according to their distance in the used log-distance model), the algorithm fills one slot at a time. For each new slot, the first transmission that is added is the not yet scheduled transmission pair with the highest link gain. Then, for each transmission that is not yet scheduled, this transmission is temporarily added to the slot. In this temporary slot, all SINRs are calculated and if all SINRs are above a specified threshold, the worst SINR that occurs in this slot is stored. The transmission that yields the largest of these worst SINR values is added to the slot. If no transmission fits in the slot, a new slot is added.

The schedule can be computed for both, the SINR model as well as the bi-directional SINR model using this approach. We call the algorithms GreedySINR (GS) and GreedyBiSINR (GBS), respectively.

4.3.2. GreedyBuffer

In this algorithm, the transmission pairs are again processed in order of decreasing link gain and one slot is filled after the other. For each transmission that is temporarily added to the slot, the buffer, which is the sum of differences between the SINR of each receiver and the SINR threshold, is calculated. The transmission that allows the largest buffer is chosen. If, for a transmission in the temporary time slot, the SINR value is less than the threshold, this transmission is discarded regardless of the calculated buffer.

We call this algorithm GreedyBuffer (GB) and the one using the bi-directional SINR model GreedyBiBuffer (GBB).

4.4. Experimental Setup

In this section, we describe the setup that is used for the scheduling experiments. We will first introduce the general wireless setup that is used in our experiments. Afterwards, the assumptions made towards the scheduling algorithms are described in Section 4.4.2. Section 4.4.3 specifies the modification to ns-3 that were necessary to realize TDMA scheduling.

4.4.1. General Wireless Setup

For wireless communication, the IEEE 802.11a standard is used. This allows communication with a data rate of up to (theoretically) 54 Mbit/s in the 5 GHz band. However, communication using high data rates such as 54 Mbit/s requires a higher SINR than communication with lower data rates. This is due to the lower redundancy used for high data rates. Since those high data rates can only be achieved for rather local communication, we chose to use a `ConstantRateWifiManager` that sends the data constantly with a data rate of 24 Mbit/s using the `OfdmRate24Mbps` data mode. This allows communication over a distance of up to about 45 meters using our log-distance model parameters. To ensure that communication is possible, we use 40 meters as maximal distance in our experiments.

We use both, the UDP and TCP protocols. As there are various TCP implementations in ns-3 to choose from, we use the `TcpNewReno` as this is the most contemporary implementation.

The propagation loss is modeled according to the log-distance model (`LogDistancePropagationLossModel`): $L = L_0 + 10 \cdot \alpha \cdot \log_{10}(\frac{d}{d_0})$, where L is the path loss in dB, L_0 is the reference path loss at distance d_0, α is the attenuation coefficient and d is the distance between sender and receiver. The parameters used are $L_0 = 46.6777, d_0 = 1$ $\alpha = 3$, which corresponds to an average free space environment with some obstacles. An attenuation coefficient α of 3 rather than 2 has been reported to be more representative of real environments [BBS06a].

To model the physical medium, the model provided by `YansWifiPhyHelper::Default` is used. The wireless net device is configured as an ad hoc station (`AdhocWifiMac`) without Quality of Service (QoS) (`NqosWifiMacHelper::Default`). Each device is assigned one distinct IP-Address within the subnet. For our simulation usually less than 255 nodes were used and therefore the 10.1.1.0/24 subnet was sufficient.

Since mobile nodes have not been considered in our experiments, a constant position model is used (`ConstantPositionMobilityModel`). Random effects such as shadowing or small scale fading (cf. Section 2.3.2) can be realized in ns-3, however, due to the running time of simulations with ns-3, which is up to a few hours without such random effects, we refrained from using those models. The introduction of random effect would further increase the number of runs needed for the results to even out sufficiently.

4.4.2. Scheduling Setup

For a theoretic analysis of scheduling algorithms, usually only the schedule length is considered. Using only the schedule length, however, is not practical for a simulation-based analysis. If one transmission did not finish in one time slot but has almost finished, accounting for another time slot is not fair. However, since we had to fall back on higher protocols to implement TDMA schedules, very small time slots are not possible (due to small but existent queues at lower layers as well as delays across layers). Thus, a compromise had to be found. After an initial examination, we settled for a time slot length of 0.1 second and an amount of data that can be transferred using 30 time slots. For the used constant bit rate, experiments have shown that one transmission can achieve up to about 130kB in one time slot (depending on the distance/signal strength). Hence, the data that must be transferred has been chosen to be 3.8 MB.

The wireless nodes are randomly placed on an area of certain dimensions. To achieve that senders and receivers can communicate directly, we calculate the senders' positions first and distribute the receivers within a certain distance around the senders. The distance d of each sender-receiver pair is uniformly drawn between a minimal and a maximal distance from the sender. It is ensured that both the sender and the receiver are place inside of the boundaries of the used area.

4.4.3. Parameters and Modifications

Since there is no TDMA scheduling support built into ns-3, some modifications of default values and minor modifications of the ns-3 source-code were necessary.

Based on the SINR value during reception of a packet[1], a packet error rate is calculated. By comparing this error rate to a uniformly drawn random value, it is decided whether the packet is successfully received or not. Even with a SINR value above the threshold, there may be some packets lost, or, with a very good SINR value, more packets than expected may be received. Hence, it may occur that a transmission needs more time slots than assigned and consequently there may be time slots that finished their transmissions before others.

The `OnOffApplication`, which is used to generate the traffic on the source nodes, by default generates traffic either all the time or according to a static schedule. As some slots may be processed completely before others, we need to enable the application to react on changed schedules. This is done by enabling the `OnOffApplication` to start for one time slot at an arbitrary time, by adding the method `StartFor(startTime, duration)`.

The transport and the link layer usually implement buffers or queues, to allow applications to give all data that must be transmitted to the lower layer at once. However, to implement scheduling using the application layer, this behavior is not desired. But, since these buffers and queues are necessary, they can not be completely disabled. Therefore, we must try to minimize the effect of those queues and buffers. As the application layer is configured to send with a constant bit rate, the queues may be small since they are constantly refilled.

In our setup, we allowed the transportation protocols to buffer two packets. To achieve this, we modified two default parameter values, namely the `TcpSocket::SndBufSize` for TCP and the `UdpSocket::RcvBufSize` for UDP such that they allow a maximum of two packets. The MAC layer implements another queue, the `WifiMacQueue::MaxPacketNumber`. This queue has also been set to 2 packets.

With those adjustments, there are at most 4 packets in the queues and buffers of the various layers. Hence, once the application stops passing packets to lower layers, the lower layers run out of packets and stop transmission.

Using TDMA schedules, the senders do not have to wait for a mechanism such as the CSMA/CA that determines whether the medium is free or whether other stations are

[1] Actually, ns-3 accounts for varying SINR values during the reception of a packet to calculate its packet error rate

sending at the moment. To achieve this behavior using the ns-3 network simulator, we need to adapt the thresholds that are accountable for the CSMA/CA behavior, namely the energy detection threshold and the Clear Channel Assessment (CCA) threshold, such that they allow to send even though other stations may be sending at the same time. This is done by setting both thresholds to a signal strength that is 5 dBm lower than that of the intended partner. Note that this does not necessarily achieve the correct behavior for schedules that are not computed according to the bi-directional SINR model.

For TDMA schedules that do not guarantee a relatively high SINR threshold, there have been some issues with the ARP-cache (which has a default DeadTimeout of 100 seconds). This timeout has been set to 0.9 seconds such that usually only few time slots are lost if address resolution fails.

4.4.4. Testing Environment

The experiments were run on a 48 core machine, consisting of 4 AMD Opteron(tm) Processor 6172 CPUs with 2.1GHz each, with a total of 256GB main memory. The machines OS is SUSE Linux 11.3-64 using Linux Version 2.6.34.10-0.6-desktop. The compiler used by the waf build system is g++ (SUSE Linux) 4.5.0 20100604 [gcc-4_ 5-branch revision 160292]. Experiments have been carried out on debug level "optimized" offered by ns-3.

ns-3 uses the MRG32k3a random number generator, as described in Section 3.1.2. In this chapter's experiments, we use seed 2 and gradually increasing run number starting at run 0. For each instance (i.e., each combination of scheduling algorithm and SINR threshold), 25 runs with different run numbers have been conducted. The figures in Section 4.5.1 show the median values of all runs, while the figures in Section 4.5.2 show the lower and upper quartiles with a line to indicate the median.

4.5. Experiments

The general SINR model is widely used and practically the standard model to compute SINR-based TDMA schedules. The bi-directional SINR model described in Section 4.2.2, however, accounts for all possible sources of interference.

Since the packets transmitted by receiving nodes are mostly small and the interference that affects the senders may be neglected, it is not clear how important it is to account for all possible sources of interference. We will consider this question in the next section by comparing the algorithms using the general SINR and the bi-directional SINR model.

TDMA schedules have the advantage of allowing sleep and duty cycles. However, they must also admit a sufficient throughput. Hence we compare the scheduling algorithms using TDMA and the 802.11 built-in CSMA/CA in Section 4.5.2.

4.5.1. Comparing SINR and bi-directional SINR

The general SINR model and the bi-directional SINR model differ in which nodes are assumed to generate interference and for which nodes are affected by interference. In the general SINR model, a new transmission pair may be added to the slot if the interference that is generated by all simultaneously active senders and occurs at the receivers does not lead towards an exceeding of the SINR threshold β. In the bi-directional SINR model, on the other hand, a time slot is valid if the interference that is generated by both, the sending and the receiving nodes, does not lead towards an exceeding of the SINR threshold β at any active node.

To compare the two interference models, we introduced each algorithm using the the bi-directional variant using the bi-directional SINR model and the standard variant, using the general SINR model.

We consider three measures of each algorithm:

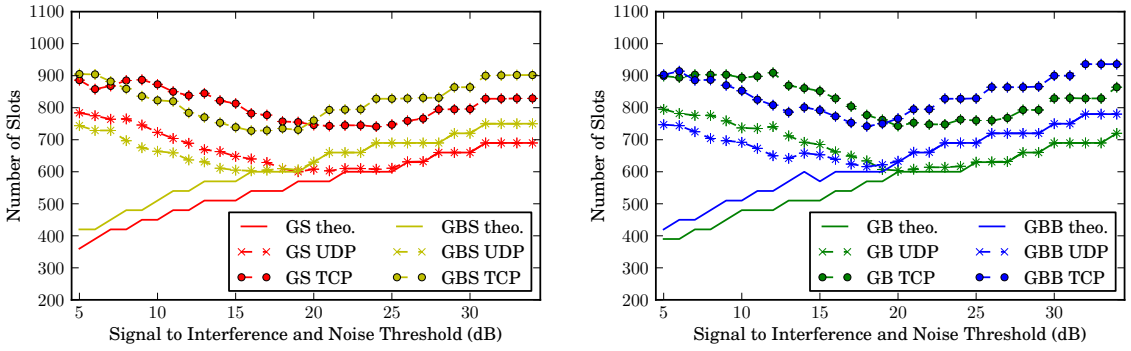
- Theoretical: This is the number of time slots that must be accounted for assuming each transmission pair needs 30 time slots to finish. For this measure we use simulations solely based on the theoretic SINR model, not on simulations with the network

simulator ns-3. Accordingly the number of slots equals 30 times the length of the calculated schedule.

- UDP: In this variant the transmissions are processed by ns-3 according to the TDMA schedule using the UDP transport protocol. The number of slots equals the last slot that is used by ns-3 to finish the transmissions.
- TCP: This variant uses the TCP transport protocol in the ns-3 simulation instead of the UDP protocol. Again, the number of slots equals the last slot that is used for transmission.

Each time slot corresponds to a duration of 0.1 seconds as described in Section 4.4.2.

In the following, we compare the time slots required by the different algorithms to finish the transmissions for varying SINR thresholds. For the experiments that led to Figures 4.2 and Figure 4.3 we used the following setup. On an area of 200×200 meters, 80 nodes have been placed. The distance between the sender and the receiver of the transmission was randomly chosen to be between 20 and 40 meters. Note that the SINR threshold has only been varied for the calculation of the TDMA schedules. The parameters of ns-3 have not been changed in our experiments[2].



(a) GreedySINR vs. GreedyBiSINR

(b) GreedyBuffer vs. GreedyBiBuffer

Figure 4.2.: The absolute performance of the TDMA scheduling algorithms for the SINR model or bi-directional SINR model for various SINR thresholds. For each algorithm a theoretical, an UDP and a TCP variant is displayed. The theoretical variant is only based on the number of slots, while for the UDP and TCP variants the time slots ns-3 needed to process all transmissions according to the TDMA schedule is shown.

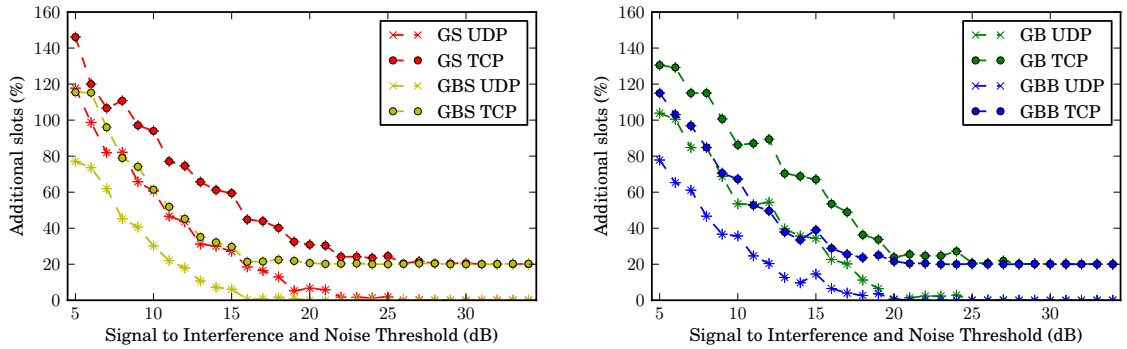
First, we can see in Figure 4.2 that, for the TDMA schedules themselves (displayed as theoretical variant) the schedule computed using the bi-directional SINR model needs about 50 slots more than the schedule computed using the general SINR model. This is due to the stricter constraints of the bi-directional variant.

Using ns-3 to process the transmissions, we can see in Figure 4.2 that the bi-directional SINR model allows the transmissions to finish slightly faster for SINR thresholds between about 10 dB and 17 dB. This also holds for the UDP transmission of even lower SINR thresholds. This is probably since we only account for the amount of data that is transmitted, packet losses do not influence the performance of UDP. For a SINR threshold below about 15 dB to 20 dB, the computed schedules can not guarantee that the messages can be decoded successfully, thus messages are lost once in a while. As more and more messages are lost for schedules computed according to lower SINR thresholds the impact of retransmission on the performance of the TCP protocol increases.

[2]except for those changes necessary to simulate TDMA scheduling, according to Section 4.4.3.

This is not surprising, as the receiving node must acknowledge the received packets not only on the link layer but also on the transport layer. Hence, it sends more packets that may interfere other transmissions. It is also not surprising that transmission of the data via TCP needs more time than transmission via UDP. This is clearly due to the overhead to ensure that each packet is received by the receiver.

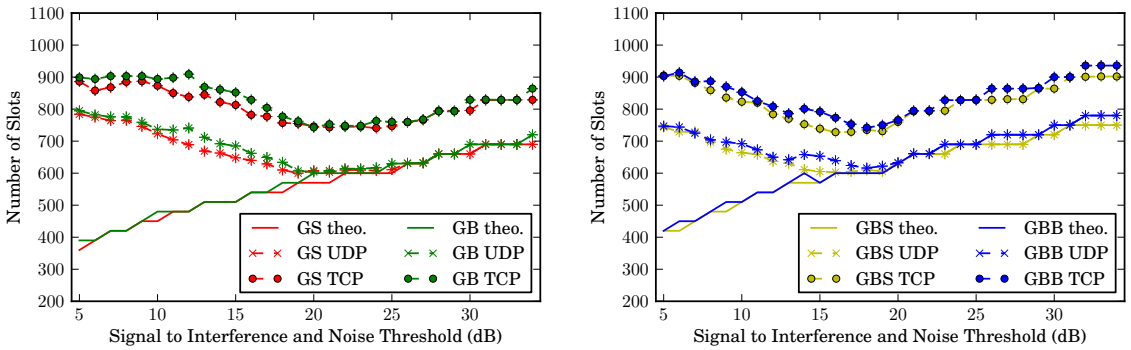
Once the SINR threshold β that is needed for successful transmission in ns-3 is exceeded, even higher values of β do not achieve more efficient schedules. As the theoretic schedule length increases, so does the number of slots needed in the simulation. In the next figure, we will be able to observe the relative overhead of the simulation of the TDMA schedules as well as this critical SINR threshold.



(a) GreedySINR vs. GreedyBiSINR (b) GreedyBuffer vs. GreedyBiBuffer

Figure 4.3.: The relative overhead of the algorithms using the ns-3 simulation over the theoretic TDMA schedule.

In Figure 4.3, we see the relative overhead of the two algorithms for both SINR variants. Both algorithms that use the bi-directional SINR model show a constant percentage of overhead after a threshold of about 19 dB is exceeded. The algorithms that use the standard SINR model settle at a similar overhead at a threshold of about 25 dB. This is probably due to the locality of sender and receiver. If the receiver achieves a SINR that is 6 dB higher than necessary to guarantee successful reception, the probability is high that the interference occurring at the sender is sufficiently small (at least for our rather small setup).



(a) GreedySINR vs. GreedyBuffer (b) GreedyBiSINR vs. GreedyBiBuffer

Figure 4.4.: Comparison of GreedySINR and GreedyBuffer and their variants GreedyBiSINR and GreedyBiBuffer.

Finally, a comparison of the algorithms themselves is depicted in Figure 4.4. We can see that for our instances of 80 nodes as well as for both transport protocols, the performance of both algorithms is mostly similar. If at all, the GreedySINR algorithm is slightly faster than the GreedyBuffer algorithm. For the bi-directional variants GreedyBiSINR and GreedyBiBuffer, the algorithms show are very similar performance for the theoretic schedules as well as for the simulation in ns-3.

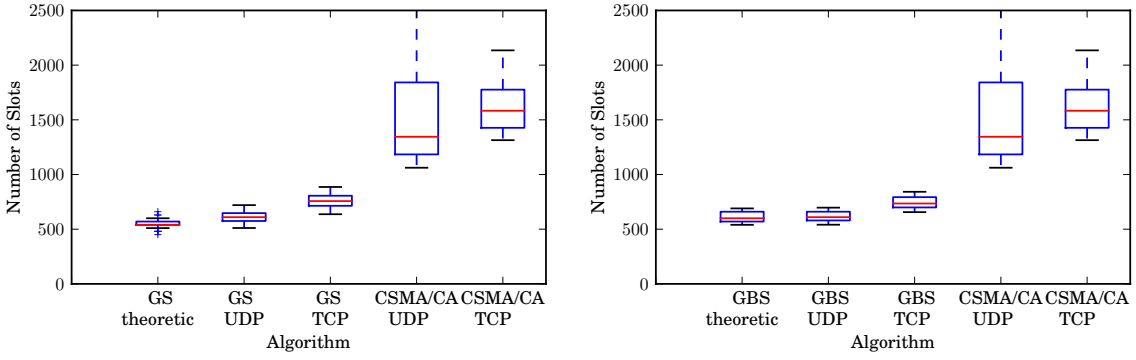
4.5.2. TDMA vs. 802.11 CSMA/CA

In order to examine the throughput-performance of the TDMA schedules in contrast to CSMA/CA, we compare the various TDMA schedules computed by the algorithms described in Section 4.3 with the CSMA/CA that is implemented in the IEEE 802.11a standard. Additionally, we consider a TDMA schedule that assigns each transmission a single slot.

As in the last section, we measure the number of time slots that are needed to transmit the data of each transmission. This is based solely on the schedule lengths for the theoretic variant and on the number of slots actually needed for the ns-3 simulation using the according transport protocol for the UDP and TCP variant. We use an area of 200×200 meters and 80 randomly placed nodes, which equals 40 transmission pairs on this area. Again, the distance between senders and receivers was uniformly distributed between 20 and 40 meters.

The CSMA/CA behavior has been achieved by creating a TDMA schedule with only one time slot. Since all transmissions are allowed to send in every time slot, only CSMA/CA or other built-in mechanisms control the medium access. Note that the energy detection threshold in ns-3 has not been increased in this case (i.e., not as described in Section 4.4.3). Such an increased energy detection threshold disables CSMA/CA and would result in uncontrolled medium access.

The results of our experiments are depicted in Figure 4.5. The experiments have been conducted for both the UDP and the TCP protocol. Such a distinction is interesting since TCP has a built-in congestion control that may react on a lossy channel as a result of high interference.



(a) GreedySINR (b) GreedyBiSINR

Figure 4.5.: The absolute performance of the TDMA scheduling algorithms (using a SINR threshold of 18 dB) compared to the CSMA/CA mechanism.

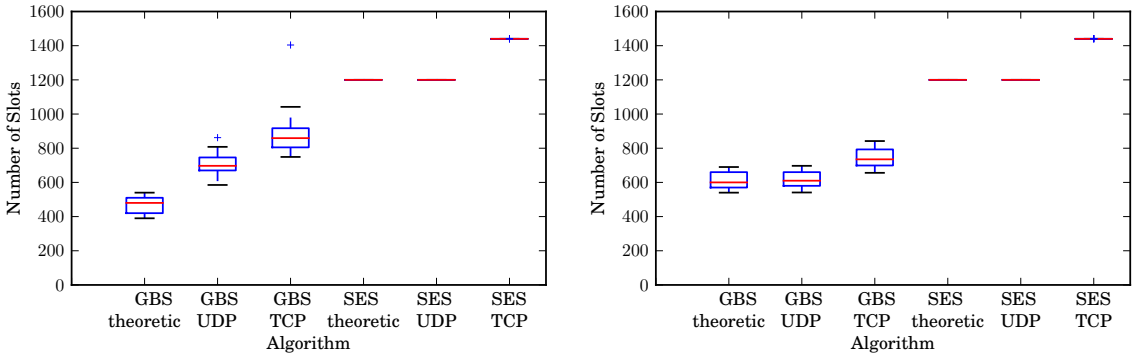
We can observe that for both, the UDP protocol and the TCP protocol, the CSMA/CA-based transmission is slower. For both protocols the time the transmissions needed to finish is about twice as high compared to the TDMA schedules if the medium access is CSMA/CA-based. This is reasonable since for our setup on an area of about 200×200 meters usually only one transmission can send in the CSMA/CA-based medium access, since for most senders at least the sender of another transmission is within the range

of about 160 meters. In the used log-distance propagation loss model with attenuation coefficient 3, the distance of about 160 meters is needed for the signal to fade such that its energy is below the energy detection threshold of the nodes in ns-3. If a signal is received with an energy above the energy detection threshold, CSMA/CA assumes that the medium is busy and postpones the transmission of packets. This does in general avoid interference, however it also introduces problems like the hidden station and the exposed station scenarios we described in Section 2.3.6. Those scenarios describe situations in which the sender does either not send even if the receiver would be able to successfully decode the sender's signal or the sender sends even though the receiver is interfered by other stations.

Such scenarios are not possible for the TDMA schedules since the SINR equation holds for each transmission. On this relatively small setup we can achieve slots with up to 7 simultaneous transmissions even for common transmission powers. On average about 2 to 3 transmissions are in one time slot[3].

This can also be seen if we compare the throughput that has been achieved. Using the median number of time slots, we can estimate a throughput based on the data which has been transmitted by all senders and the time that has been needed to finish the transmission for each pair. This throughput is slightly above 40 Mbit/s for the TDMA-based transmission while it is about 20 Mbit/s for the CSMA/CA-based transmission.

After considering a special TDMA schedule that features only one slot, we are now considering the other extreme, a TDMA schedule that features one slot for each transmission. We will call this schedule *separate schedule* (SES). The separate schedule is basically a scheduling algorithm that is too cautious or assumes a too high SINR threshold and hence uses a separate slot for each transmission. In these experiments we study the effects of a too low SINR threshold by comparing the schedule computed by the GreedySINR and the GreedyBiSINR algorithms to the separate schedule. The SINR threshold of 8 dB, which is too low to guarantee successful reception, and the SINR threshold of 18 dB, which is high enough to guarantee successful reception (for the considered bi-directional scheduling algorithms) are used.



(a) SINR threshold of 8 dB

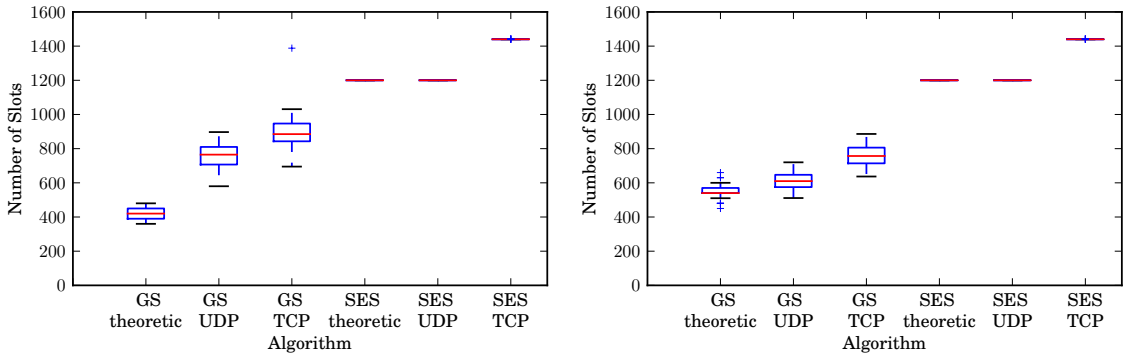
(b) SINR threshold of 18 dB

Figure 4.6.: Comparing the TDMA schedules computed by the GreedyBiSINR algorithm to the separate schedule, a TDMA schedule that assigns each transmission its own time slot. The SINR threshold for the scheduling algorithms is set to 8 dB on the left and 18 dB on the right.

In Figure 4.6 the results of this experiments are depicted for the GreedyBiSINR algorithm. We can see that even for the lower SINR threshold, the computed schedule is faster

[3]Note that for distances of almost 40 meters between sender and receiver, some transmissions can hardly be combined with any others on the area of 200×200 meters due to the high SINR required for 24 Mbit/s transmission.

than assigning each transmission a separate time slot and hence enable strong transmission without any interference. This effect is probably due to some relatively strong transmission that are able to finish fast even though other transmissions are sending and then stop emitting signals. Hence the SINR for some more critical transmissions (regarding the achieved SINR) is increased and they are now able to successfully transmit messages. For the SINR threshold of 18 dB, the simulations conducted with ns-3 does not require more than the time slots accounted for. This implies that the SINR threshold of 18 dB is sufficient for the algorithms that use the bi-directional SINR model, it may even be slightly too cautious (cf. Figure 4.2). The SES algorithm does not show show any deviation from the median. This is since the number of transmissions is fixed and hence the number of time slots is fixed as well. Since each transmission has its own time slot there is no interference at all and each transmission can sent the anticipated amount of data in each time slot.



(a) SINR threshold of 8 dB

(b) SINR threshold of 18 dB

Figure 4.7.: Comparing the TDMA schedules computed by the GreedySINR algorithm to TDMA schedules that assign each transmission its own slot. The SINR threshold for the scheduling algorithms is set to 8 dB on the left and 18 dB on the right.

In Figure 4.7, the results for the GreedySINR algorithm are depicted. The results are similar to those for the SINR threshold of 8 dB, but for the SINR threshold of 18 dB, we can see that the overheads of the schedules computed by the GreedySINR algorithm are higher. This implies that for the general SINR variant of the algorithm the SINR threshold must be chosen larger to achieve transmissions without interference.

These experiments have also been conducted with the GreedyBuffer and the GreedyBiBuffer. We conducted similar experiments with the algorithms GreedyBuffer and GreedyBiBuffer. The results of these experiments were very similar to the ones that we just presented. The according figures are given in the Appendix A.2.

4.6. Discussion

The experiments in Section 4.5.1 showed that the performance of the considered scheduling algorithms in our simulations is improved for certain SINR thresholds if the bi-directional SINR is used instead of the general SINR. As the considered algorithms are rather simple, it would be interesting to examine how TDMA schedules computed by more elaborate algorithms perform using the bi-directional SINR model to decide whether a transmission pair can be added to a time slot.

The comparison of TDMA schedules and the IEEE 802.11 CSMA/CA in Section 4.5.2 showed that we can achieve about twice the throughput by using TDMA schedules instead of the CSMA/CA mechanism. Hence, TDMA schedules do not only enable the use of an energy-efficient communication due to sleep and duty cycles, which may be used to conserve energy, but also by enabling a considerably higher throughput.

In this chapter, we considered scheduling algorithms and assumed that the transmissions are given. In the next chapter, we consider topology control which, given a wireless network, selects a subset of all links with the goal to minimize interference by restricting communication to this subset of links. A combination of both scheduling and topology control is considered in Section 5.5.

5. Topology Control

In recent years, topology control has been a major field of research in wireless sensor networks, and several algorithms for the computation of network topologies have been proposed. In this chapter, we study algorithms that are so far mainly theoretically analyzed and which often provide guaranteed graph properties such as constant maximum vertex degree, constant energy or distance stretch factors. The wireless sensor network (WSN) is modeled with a graph, which is assumed to be connected (i.e., each node in the underlying WSN can communicate with all other nodes using multi-hop communication).

The problem of topology control can be seen as a selection of edges such that certain desired properties can be achieved. Using the notation of a graph $G = (V, E)$, a subgraph $G' = (V, E')$ with $E' \subset E$ must be computed such that, for example, efficient communication, a sufficiently high throughput and a maximized network lifetime can be achieved. Of course, depending on the actual application the desired properties may differ. Several quality criteria that are commonly considered are described in the next section along with a brief overview over topology control. Afterwards, in Section 5.2, we introduce the algorithms examined in this thesis. The setup for the topology control experiments is described in Section 5.3, followed by the experiments itself in Section 5.4. In Section 5.5, we introduce methods to measure the performance of the topologies considered in this chapter using the TDMA scheduling algorithms described in the previous chapter. Experiments and results considering this performance measure are described in Section 5.6. The chapter is concluded with a discussion in Section 5.7.

5.1. Introduction to Topology Control

Like most algorithms for WSNs, topology control aims at minimizing energy consumption and hence extending the network lifetime while conserving (or achieving) good overall network performance. To achieve this, there are mainly two options. First, minimizing the energy that is is consumed for each transmission. This can be done by preferring short connection links above longer ones and adjusting the transmission powers accordingly. And second, minimizing the interference and hence making the transmission process itself more efficient.

While it is relatively clear how links can be chosen such that a small transmission power suffices, it is not so clear how the interference can be minimized. In the algorithms that are considered in this chapter, it is often assumed that a low node degree implies low interference. However, this is not necessarily the case; in fact, it is argued in [BvRWZ04] interference is not effectively constrained by topology control algorithms that assume interference is minimized by a low node degree.

As our analysis is based on simulations conducted with the network simulator ns-3, we are not constrained to theoretically modeling and analyzing interference, but we can use ns-3 to analyse the practical performance of the topologies. This allows us to draw conclusions regarding the influence of several properties like sparseness, vertex degree or stretch factors on the achievable network performance.

In this thesis, it is assumed that communication between the nodes is responsible for major parts of the energy consumption. For graph-theoretic models, it is usually assumed that the energy that is needed for communication between two nodes u and v is given by a power law

$$\text{energy}(u, v) := \text{dist}(u, v)^{\alpha},$$

where α is called the *attenuation coefficient* which is usually between 2 and 5 (see Section 2.3.2). In order to conserve energy, the links must be chosen such that the transmission power can be reduced. In practice, however, the links must not only have a low transmission power but also yield high throughput even in lossy environments. Hence, the transmission power must not be too low.

Several quality criteria have been proposed to measure the quality of network topologies. In the following, we give a brief overview, which is based on [WW07].

Connectivity: Usually each node must be able to communicate with any other node. Hence, connectivity of the computed topologies is a basic requirement for topology control algorithms. Sometimes one even asks for *k-vertex-connectivity* or *k-edge-connectivity*. This ensures that at least k vertices or edges must be removed until the graph is no longer connected.

Symmetry: Topologies that allow communication along a link in only one direction are not desired. With unidirectional links, communication gets more complex even for supposedly simple messages like ACKs. Most routing algorithms require bi-directionality of the communication links.

Stretch factors: Considering the original graph G and the subgraph G' constructed by the topology control algorithm, the stretch factor is the largest ratio of a value in G' compared to the value in G. This ratio can be calculated for different measures such as the hop-distance, the Euclidean length of shortest path, or the energy consumption along an energy-minimal path. Those measures yields different stretch factors like the *hop stretch factor*, the *distance stretch factor* or the *energy stretch factor*. The distance stretch factor, for example, is the largest ratio of the Euclidean distance of a shortest path connecting two nodes in G' and the shortest path connecting the same nodes in G.

If G' has a constant distance stretch factor c, we say that G' is a *c-distance spanner* of G. The same applies for energy and hop spanners. If the attenuation coefficient α is larger than 1, a constant distance stretch factor also implies a constant energy stretch factor [SWL04].

Sparseness: To ensure simplicity and maintainability of the network, a topology control algorithm should yield sparse graphs. A graph is sparse if it has few edges, i.e., its number of edges is constant in the number of nodes. This yields a constant average node degree. The node degree, itself, however may be linear in the number of nodes. Hence, a constant maximum node degree is a stronger requirement. Routing algorithms are usually more energy efficient on sparse networks, due to the less complex layout.

Additionally, it is often assumed that sparse graphs minimize the interference of topologies. However, as mentioned earlier, there is no consistent argumentation for this assumption.

Throughput: The algorithm should not limit the throughput of a network. In theoretic models, the throughput is often measured by using a *bit-meter* metric. A bit-meter equals one bit, transported over a distance of one meter in the network. By calculating the maximum bit-meter per second that the network can transport, the throughput of the network can be determined.

Interference: Signals from a specific sender do not only reach the intended receiver, but also any other node that is within range. For those unintended receivers, this signal does not provide valuable information but hinders the reception of other signals. Signals that hinder other transmissions are called interference (cf. Section 2.3.5). As mentioned, topology control aims at reducing interference.

Adaptability: Since wireless sensor networks are often deployed in the environment and may even be mobile, a topology should quickly adapt towards changes in the network. The adaptability of a topology can be measured in how many network nodes must be updated if a node moves or is switched off.

Planarity: Some geometric routing algorithms require the graph to have an embedding in the plane without crossing edges. This is called planarity.

Within the scope of this thesis, we will consider mostly algorithms that yield connected topologies. Topology control algorithms that compute not necessarily connected topologies are only considered if the resulting topologies are connected for most instances. As IEEE 802.11 relies on symmetric links (due to MAC layer ACKs), we require the topologies to be symmetric. In Section 5.4, we conduct experiments concerning the performance of topologies with various stretch factors as well as different levels of sparseness. We do not calculate the bit-meter of the networks, but take a rather practical approach by using random sender-receiver pairs that need to transport data through the network to compare the throughput.

Reducing the interference is one of the main motivations for topology control. However, by restricting nodes to communicate with only some of their neighbors, topology control does only decrease the logical degree of the nodes. The physical degree has not changed and hence the interference is not reduced. To reduce the interference from one node on another node, the transmission power of the sending node must be reduced. For common transmission powers, it is argued in [NKSK02] that decreasing the transmission power such that it is on the critical value for connectivity is the best solution. For variable transmission power minimizing interference is more complex. For two-dimensional networks, for example, it has been shown that minimizing the maximum interference[1] is NP-hard [Buc08].

In this thesis, we will not consider minimizing interference as a separate problem. Since ns-3 simulates the impact of interference, the effect of interference is automatically included in the performance of the topologies.

5.2. Algorithms

There exist various algorithms in the field of topology control. Due to the focus of this thesis, we consider algorithms that produce topologies with certain graph-properties, e.g., distance spanners. This includes the following topologies: EMST, a spanning tree with minimal overall edge weight. XTC, which is based on neighborhood relationships and Gabriel Graph (GG), which is based on exclusion regions. The Yao Graph (YG) connects each node to neighbors in different directions. The graph with all possible communication links which we call the All Links Graph (ALG), the Restricted Link Strength Graph (RLS) which uses only communication links whose signal strength is higher than a certain threshold as well as hop, distance and energy spanners are considered.

5.2.1. All Links Graph (ALG)

In the All Links Graph (ALG), all possible communication links are represented by an edge. All other topologies are subsets of this graph. This graph sometimes is called max-power-graph in the literature.

As we assume the wireless sensor network to be connected, this graph is connected as

[1] The receiver interference is modeled as the number of transmission ranges in which the receiver lies; the transmission ranges are modeled as disks.

well. It is not necessarily planar or sparse and has a maximum vertex degree of $n-1$. Since energy and distance stretch factors are relative to this graph, both factors are 1. A simple algorithm to construct the all links graph is given in Algorithm 5.1. After receiving all messages, the communication links for each node are in the set E.

Algorithm 5.1 ALL LINKS GRAPH (for each node $u \in V$)

1: Send a broadcast
2: **for each** incoming messages from some node v **do**
3: add the edge (u, v) to E

5.2.2. Euclidean Minimum Spanning Tree (EMST)

A minimum spanning tree is a connected graph with minimal overall edge weight. If the edge weights are the Euclidean distances between the nodes, this spanning tree is the EMST. It is planar, sparse and has a maximum vertex degree of 6. It is a tree and hence there is only one path connecting two nodes. This path may be long even for nodes that may be physically able to communicate directly. Therefore its distance and energy stretch factors are $n-1$ (cf. Table 5.1).

The EMST can not be constructed locally, but for $k \geq 2$ there exists a k-localized distributed algorithm, which yields a total weight of less than $1 + \frac{2}{k-1}$ times the weight of the optimal solution [KPX07]. Local Minimum Spanning Tree (LMST), a localized algorithm to compute an approximation of the EMST, still produces a connected, planar and sparse tree with maximum node degree of 6 [LHS05].

In this thesis, however, we consider the globally constructed EMST-graph: G_{EMST}. Localized algorithms try to approximate the global algorithms and the main properties of the EMST also hold for the localized algorithms. We account for some sub-optimalities that may occur during locally constructing the EMST by using a integer-based EMST.

Algorithm 5.2 EMST

Input: C is a set of trees (forest), initially containing all vertices of the graph,
S is a set containing all edges of the graph
1: **while** $S \neq \emptyset$ and C is not spanning **do**
2: extract edge $e \in S$ with minimum weight
3: **if** e connects two trees in C **then**
4: add e to the forest C {i.e. join the two trees together}

5.2.3. Relative Neighborhood Graph (RNG)

The *relative neighborhood* of an edge is the area that is closer to the two adjacent nodes than the distance between the nodes. An edge (u, v) exists in this graph if the relative neighborhood of this edge is empty, i.e., if $d(u, v) \leq \min_{w \in V \setminus \{u,v\}}(d(u, w), d(v, w))$ using Euclidean distances. The relative neighborhood of an edge can also be seen as the intersection of the disks centered at the two adjacent nodes with radii equal to the distance of the two nodes.

5.2.4. XTC

Whether edges are in the XTC graph or not is based on relative distances between the adjacent nodes. If the distances are Euclidean, XTC computes a subgraph of the Relative Neighborhood Graph (RNG). The XTC algorithm has been proposed in [WZ03].

Prior to computing the topology, a strict order must be chosen. We use the Euclidean distance, but if the distance is unavailable, the link qualities or other similar orders may be used. First, node u must establish the strict order $\prec_u$ over its neighbors and exchange the order with his neighbors. Then, u can process its neighbors according to the order $\prec_u$.

	ALG	EMST	XTC	GG	YG_6	RLS
connected	yes	yes	yes	yes	yes	no
planar	no	yes	yes	yes	no	no
sparse	no	yes	yes	yes	yes	no
maximum vertex degree	$n-1$	6	6	$n-1$	$n-1$	$n-1$
distance stretch factor	1	$n-1$	$n-1$	$\sqrt{n-1}$	$\frac{1}{1-2\sin\frac{\pi}{6}}$	-
energy stretch factor	1	$n-1$	$n-1$	1	$(\frac{1}{1-2\sin\frac{\pi}{6}})^{\alpha}$	-
spanner	yes	no	no	no	yes	no

Table 5.1.: Quality criteria of the considered algorithms [WW07, page 93].

If neighbor v is processed, an edge between u and v is added to the (final) topology if there is no common neighbor w that comes before v in u's order and before u in v's order. Symmetric edge weights are assumed and hence XTC computes an undirected graph: G_{XTC}. A pseudo-code is given in Algorithm 5.3.

For Euclidean distances, it holds that XTC is equal to the RNG if no two nodes have the same distance to a third node (i.e., the distance-based order is strict). If the distance-based order is not strict, each node may be assigned a (locally distinct) identification number to break the tie.

The XTC-graph is planar and sparse and (using the Euclidean distances) its maximum vertex degree is 6. The distance and energy stretch factors are $n-1$ (cf. Table 5.1).

Algorithm 5.3 XTC (for each node $u \in V$)

1: Establish order $\prec_u$ over u's neighbors
2: Broadcast order
3: Receive orders
4: **for all** v in increasing order according to $\prec_u$ **do**
5: **if** $\nexists w : w \prec_u v$ and $w \prec_v u$ **then**
6: Add v to final neighbor set

5.2.5. Gabriel Graph (GG)

In order to introduce Gabriel graphs, we need the concept of disks.

Definition 5.1. The disk(u, v) is the closed disk with diameter $d(u, v)$ containing the nodes u and v.

In the Gabriel Graph (GG), there is an edge between two nodes u and v, if and only if the disk with diameter $d(u, v)$ between the nodes does only contain u and v.

To construct the topology based on the Gabriel graph, a local algorithm (see Algorithm 5.4) is executed on each node u in the network. First, the position information is broadcasted to all neighbors. For each message that is received from a neighbor v, the corresponding edge is added to the set of all edges incident to u ($E_G(u)$). If there is no (so far discovered) edge (u, w) with $w \in \text{disk}(u, v)$, the edge (u, v) is added to the temporary set of edges of the Gabriel Graph ($E_{GG}(u)$). Then, for each edge (u, w) that has earlier been added to the edges of the Gabriel Graph, it must be confirmed that the disk is still empty. If v is in disk(u, w), this is not the case and (u, w) must be removed from the temporary set $E_{GG}(u)$.

Once all messages have been processed, the set $E_{GG}(u)$ is no longer temporary, but contains all edges of the Gabriel Graph incident to u. $G_{GG} = (V, E_{GG})$ with $E_{GG} = \cup_{u \in V} E_{GG}(u)$.

Algorithm 5.4 GABRIEL GRAPH (for each node $u \in V$)

1: $E_G(u) = \emptyset$ {the set of all edges incident to u}
2: $E_{GG}(u) = \emptyset$ {the set of edges incident to u in the Gabriel Graph (which is temporary during construction)}
3: Send a broadcast with position information
4: **for each** incoming messages, from some node v **do**
5: add (u, v) to $E_G(u)$
6: **if** there is no $(u, w) \in E_G(u)$ such that $w \in \text{disk}(u, v)$ **then**
7: add (u, v) to $E_{GG}(u)$
8: **for all** $(u, w) \in E_{GG}(u)$ **do**
9: **if** $v \in \text{disk}(u, w)$ **then**
10: remove (u, w) from $E_{GG}(u)$

5.2.6. Yao Graph (YG)

The Yao graph divides the surroundings of each node in c cones of equal angle and adds edges only to the nearest neighbor in each cone. If there are two or more nearest neighbors, one can break the tie by choosing one neighbor arbitrarily or adding an edge to all nearest neighbors. The former was chosen by the original authors [Yao82] and is used in this thesis.

To compute the Yao Graph locally, each node $u \in V$ must broadcast its position to all neighbors. For each message that is received at u from a sender v the cone v is in is calculated, Then the distance between u and v is compared to the distance of the nearest neighbor in this cone that has so far been found. If v is the nearest neighbor that has so far been considered in cone i, $n_u[i]$ is set to the edge (u, v). For each node u, the selected edges are given in $n_u[i]$ after the algorithm finished. A pseudo-code of the algorithm is given in Algorithm 5.5.

Note that the Yao Graph as described does not yield a symmetric graph. An uni-directional edge from u to v in the Yao Graph does not imply that there is another uni-directional edge from v to u. The Yao Graph may even be only weakly connected, which means that it is connected only if the links are considered to be bi-directional. The bi-directional Yao Graph is called *undirected Yao Graph*.

As the IEEE 802.11 standard relies on bi-directional communication links, we use the undirected Yao Graph in the remainder of this thesis. The YG_c divides its surrounding in c cones. The YG_c can be constructed for different values of $c \geq 6$. In this thesis, we use $c = 6$. The YG_6 has a maximum vertex degree of $n - 1$, a hop stretch factor of $1/(1 - 2\sin\frac{\pi}{6})$, and an energy stretch factor of $(1/(1 - 2\sin\frac{\pi}{6}))^{\alpha}$ (cf. Table 5.1).

Algorithm 5.5 YAO GRAPH YG_c (for each node $u \in V$)

Input: $c \geq 6$
1: Initialize $n_u[\,]$ {an array of edges with one slot for each cone}
2: Send a broadcast with position information
3: **for each** incoming messages from some node v **do**
4: $i \leftarrow$ number of the cone v is in
5: **if** $n_u[i]$ is not set or $\text{len}(n_u[i]) > \text{len}(u, v)$ **then**
6: $n_u[i] \leftarrow (u, v)$

5.2.7. Restricted Link Strength Graph (RLS)

In the Restricted Link Strength graph (RLS), a communication link is represented by an edge if and only if its link strength is above a specified threshold. For arbitrary values of x, the graph that is restricted to links with a signal strength of x dBm or higher is denoted by $\text{RLS}_{x\,\text{dBm}}$.

Note that if the link strength is restricted too much, the RLS graph may not be connected anymore. In this case, the distance and energy stretch factors are undefined. An algorithm for the computation of the RLS graph is given in Algorithm 5.6

Algorithm 5.6 RESTRICTED LINK STRENGTH (x) (for each node $u \in V$)

Input: x {signal strength threshold in dBm}
1: Send a broadcast
2: **for each** incoming message from some node v **do**
3: **if** the signal strength of the incoming message exceeds r **then**
4: add (u, v) to E

5.2.8. Hop, Distance and Energy Spanner

A *spanner* is a subgraph of a graph with a constant ratio of a certain value in the subgraph over the same value in the original graph. For example, a *c-hop spanner* is a graph in which for each two nodes u and v it holds that each hop-minimal path between u and v in the subgraph has at most c times as many hops as a hop-minimal path between u and v in the original graph. For an attenuation coefficient $\alpha \geq 1$ it holds that if a graph G' is a distance spanner it is also an energy spanner (cf. Section 5.1).

Depending on the type of spanner that should be constructed, the metric does either count the number of hops, measure the distance of a shortest path between nodes, or calculate the energy used to send a signal from one node to another. As the metric depends on both the set of vertices V and the set of edges E of the graph, we denote the used metric by $\text{metric}_G(u, v)$ in our algorithm. We denote the original graph by $G = (V, E)$ and the spanner graph that is constructed by $G' = (V, E')$.

We use the following algorithm to construct the different c spanners: First each node sends a broadcast (with position information if the distance or energy metric is used). The edge along with position information if available is stored upon reception of the message. The temporary set of chosen edges E' is initialized as an empty set and the set of edges that represent all possible communication links is denoted by E. Then, all edges are processed in order of decreasing signal strength. For each edge it must be calculated if the ratio of the metric in the temporary graph (using the edges in E') over the metric in the full graph (using the edges in E) is larger than c. If this ratio is larger than allowed, the edge is added to the temporary set of edges E' which forms the set of edges of a spanner once the algorithm ends.

Algorithm 5.7 SPANNER ($\text{metric}_G(u, v)$, c)

Input: $\text{metric}_G(u, v)$ {hop, distance or energy metric based on the edges in E }
c {ratio that must be achieved}
1: Each node sends a broadcast
2: **for each** incoming messages [at node u, from node v] **do**
3: add (u, v) to E
4: $strength(u, v) \leftarrow$ signal strength of the signal
5: $E' \leftarrow \emptyset$
6: **while** $E \neq \emptyset$ **do**
7: $(u, v) \leftarrow (u, v) \in E$ with minimum $strength(u, v)$
8: **if** $\frac{\text{metric}_{G'(u,v)}}{\text{metric}_G(u,v)} > c$ **then**
9: add (u, v) to E'

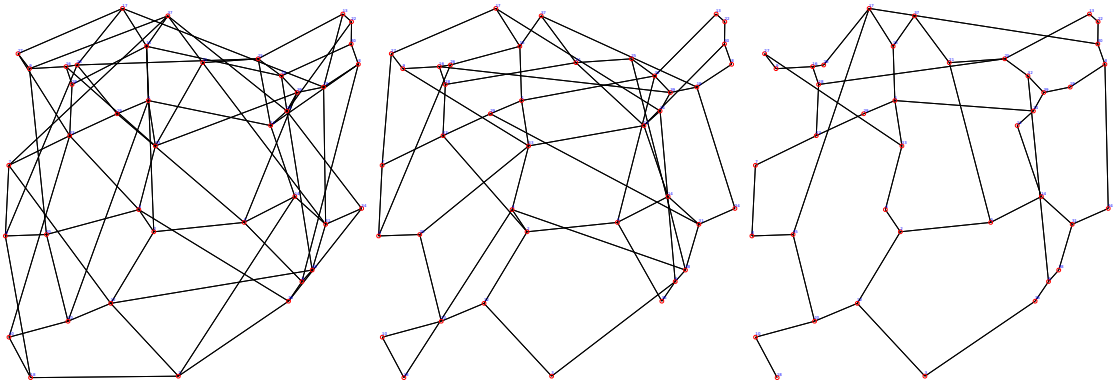
The algorithm described is only one way to construct such spanners. If the edges are considered in another order, other spanners may be computed. This algorithm is not appropriate for local computation as it is based on a global order of all links. However, a

localized computation of spanners is possible. An algorithm that constructs a spanner for fixed or variable transmission powers is described in [PR10].

Note that calculating the metric-function $\text{metric}_G()$ is trivial for the hop and distance spanner. For the energy spanner, a breadth-first search using the edges in E is sufficient.

5.2.9. Visual Comparison

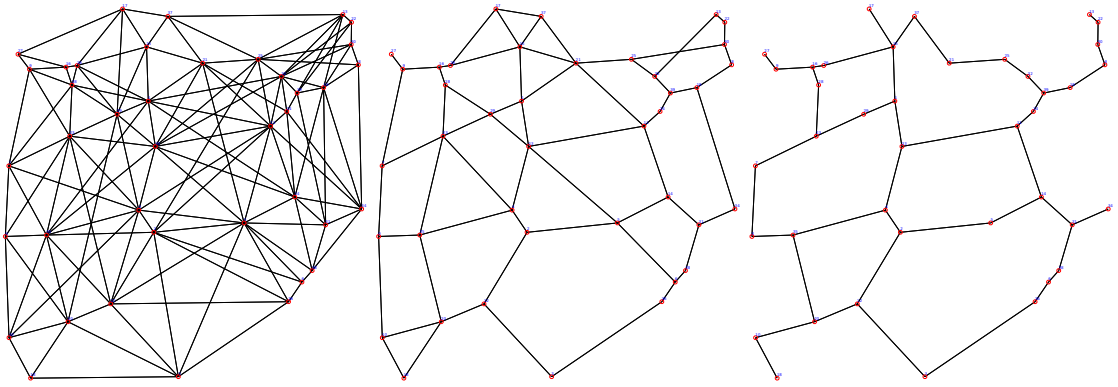
In order to gain a first impression of the topologies computed by the described topology control algorithms, we compare the topology computed for a random network visually in this section. Small dots visualize the position of a wireless sensor node while for each communication link that is selected by the topology a black edge is drawn between the sensor nodes. We used 40 nodes on an area of 200×200 meters for this visual comparison.



(a) 2-hop spanner (b) 3-hop spanner (c) 5-hop spanner

Figure 5.1.: Visual comparison of different hop spanners.

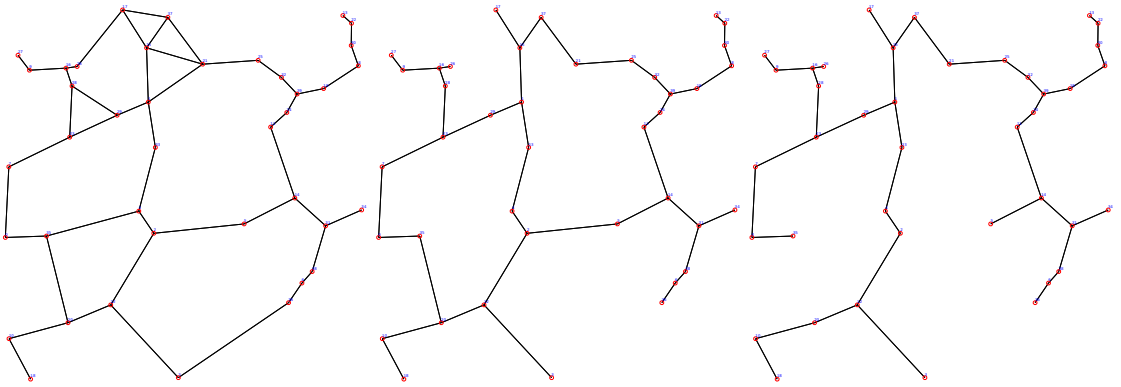
Before comparing all considered topologies, we consider the topologies constructed for different parameters c of the spanner algorithm. In Figure 5.1, the various c-hop spanners are depicted. We can see that the as the parameter c decreases more long communication links are added between nodes that are relatively far away. The underlying structure which ensures connectivity remains mostly the same. For the distance spanners depicted



(a) 1.1-distance spanner (b) 1.5-distance spanner (c) 2.5-distance spanner

Figure 5.2.: Visual comparison of different distance spanners.

in Figure 5.2, the amount of long links that are added is lower. Only very few links in the 1.1-distance spanner are as long as some of the links in the 2-hop spanner. We can also see that the density of the network seems relatively homogeneous. The energy spanner shown in Figure 5.3 are considerably less dense. This is due to the cubically increase of energy consumption with the communication distance. Hence, a longer detour using shorter edges is often more energy efficient that the direct communication.



(a) 1.1-energy spanner (b) 2-energy spanner (c) 5-energy spanner

Figure 5.3.: Visual comparison of different energy spanners.

In Figure 5.4, the toplogies computed by all considered algorithms are depicted. We can see that all topologies reduce the density considerably by comparing them to the ALG topology depicted in 5.4(a). Both the 1.1-distance spanner and the Yao graph are relatively dense, however, the possible communication links are distributed relatively homogeneous over the network. The EMST, XTC and the GG are rather sparse topologies. The EMST has clearly the lowest number of edges, followed by XTC and the Gabriel graph; this is in fact an inclusion relation. The 1.1-energy spanner is not in this relation, however its density seems to be somewhere between the XTC and the Gabriel graph. The visual impression of the 2-hop spanner seems relatively orderdered, with some relatively long edges spanning across large parts of the network. The RLS restricts the used communication links to relatively strong links. The distribution seems not very homogeneous over the network but depends on the node density within the network.

5.3. Simulation Setup

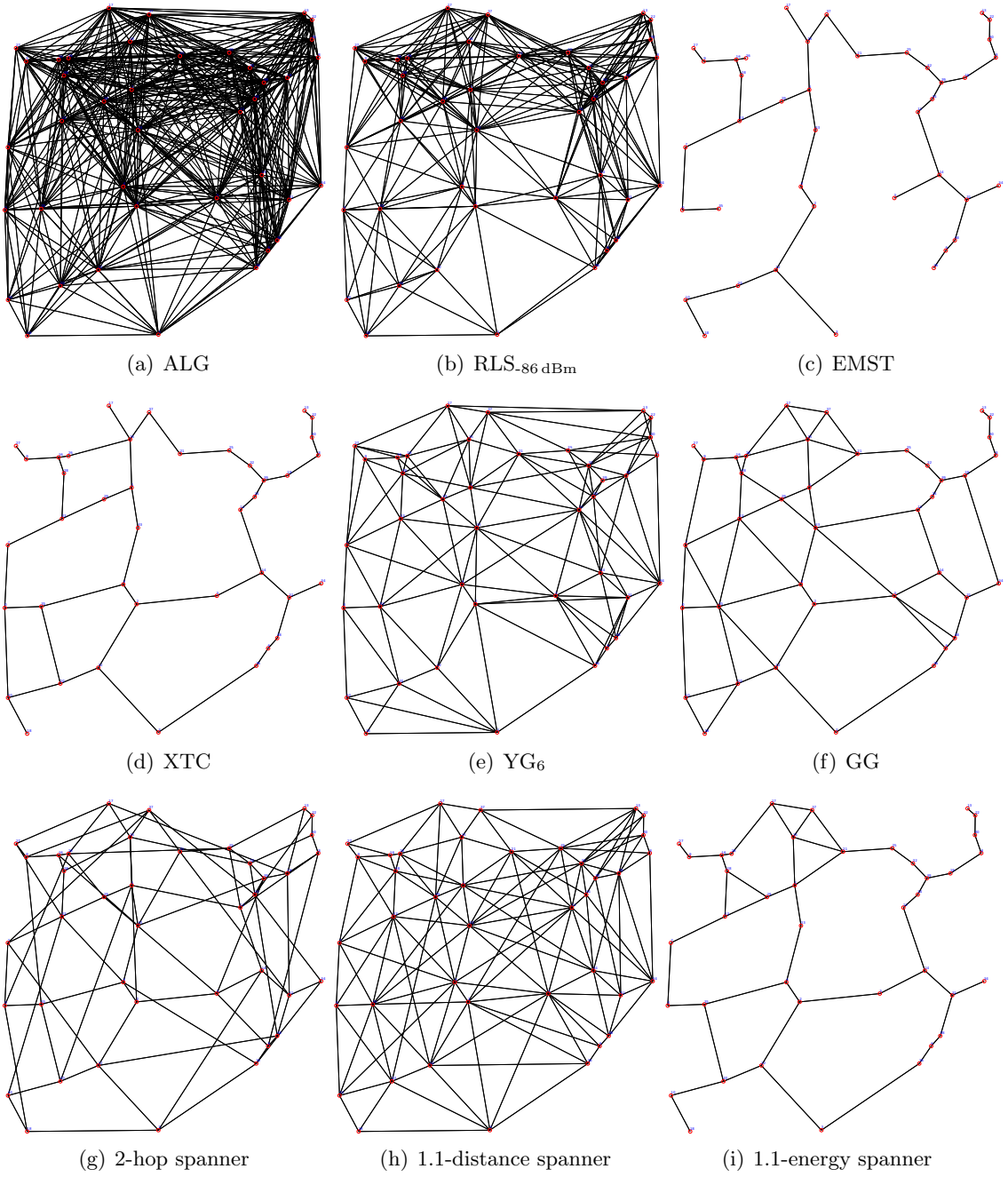
In this section, we describe the models and parameters that we used in the ns-3 simulator. For a description of the models of a standard wireless setup, see Section 3.3.1. Throughout this thesis, we use ns-3 in version 3.13, which is has been released on December 23, 2011.

As there is a description of the general setup of our wireless network in Section 4.4.1, we will only highlight the changes that apply for this chapter. In Chapter 4, a constant-rate rate-control-manager has been used to achieve better comparability. In this chapter, however, a more realistic behavior of our nodes is required. Therefore, the `IdealWifiManager` is used for rate-control and hence varying data rates based on the SINR of the communication link are used. In this chapter, we restrict the transmissions to the TCP protocol implemented by `TcpNewReno` since reliable data transfer is desired. The other parts of the general wireless setup in Section 4.4.1 also apply to this chapter. This includes the propagation loss according to the log-distance model with parameters $L_0 = 46.6777, d_0 = 1, \alpha = 3$ and the physical layer according to `YansWifiPhy::Default`.

5.3.1. Parameters and Modifications

To realize a more adequate behavior of the sending application as well as the used protocols, some adaptions to the default protocol parameters and the protocol behavior of ns-3 had to be made:

- The TCP buffer size for sending (`TcpSocket::SndBufSize`) is set to the amount of data to be transferred. Since the `OnOffApplication`, which is used for generating the traffic, does not care about loosing data, a small TCP sending buffer may get filled completely and data is lost. In this case, a reception of the required amount of data at the receiver would not be possible.



(a) ALG (b) $RLS_{-86\,dBm}$ (c) EMST

(d) XTC (e) YG_6 (f) GG

(g) 2-hop spanner (h) 1.1-distance spanner (i) 1.1-energy spanner

Figure 5.4.: Topologies that were generated for a random network of 40 nodes on an area of 200x200 meters.

- To avoid that TCP sockets are unable to establish a connection in a lossy environment, the number of SYN retransmissions is increased to 1000 (i.e., the TCP socket attempts to establish a connection even though it may have failed often). Note that ns-3 does not use an upper bound for the retransmission timer, which determines the time that is waited before a lost packet is retransmitted.

- The initial TCP retransmission timeout exactly doubles after each SYN retransmission. This exact doubling is not desired, since two senders that could not establish a connection due to interference with the corresponding receivers would both retrans-

mit at exactly the same time after the back-off[2]. Therefore the doubled retransmission timeout is multiplied by a uniformly distributed random value in the range [0.99, 1.01]. This is not a parameter that can be adjusted in ns-3, but a modification to the ns-3 code. A patch can be found in Appendix A.1.

- Since we frequently use dense instances that result in transmissions with high interference, packets are lost. If the Address Resolution Protocol (ARP) is not able to retrieve the address of a node, the node is flagged as dead for 100 seconds by default. This timeout (`ArpCache::DeadTimeout`) is decreased to 5 seconds in order to give other transmissions a short time to finish but to also give back-off control back to TCP fast.

- Additionally, the number of retransmissions of the ARP-Request (`ArpCache::MaxRetries`) is slightly increased from 3 to 5, to enable a more stable ARP behavior throughout the simulations.

The energy detection threshold in ns-3 is set to -96 dBm by default. Thus, communication links that have a signal strength of below -96 dBm are not considered strong enough to be received. To compute topologies, we need an initial set of edges out of which the topology control algorithm can select a subset. However, we do not simply consider all communication links up to the threshold of -96 dBm but give the simulator an initial phase in which each node sends 5 broadcast messages. For each node that successfully receives at least one of the messages, a (bi-directional) communication link is added to the initial set.

This set is used by the topology control algorithms to select the subset that composes the resulting topology, if not declared otherwise.

In the next section, we describe how ns-3 can be restricted to the communication links selected by the topology control algorithms.

5.3.2. Routing Algorithms and Neighborhood

A channel in ns-3 distributes signals sent by any node attached to the channel to all other nodes attached to the channel as described in Section 3.3.1. Restricting the channel would be sufficient to ensure that packets are sent only to neighbors that are chosen using the topology control algorithm. One approach that would have such an effect is adding a propagation loss model that increases the path loss of packets between neighbors that are not allowed in the topology, such that reception is impossible. Using this approach, the network can communicate only over the allowed links. However, not only the desired data signal would be restricted to the selected communication links but also interference. Since ns-3 is a packet-based network simulator, interference is received only if it is received with a packet.

This discards the propagation loss approach, since the relatively realistic modeling of interference is one of the reasons a network simulator is used in this thesis. Hence, the channel-based approach to restrict communication links in ns-3 must be more complex. The interference of packets must be considered, but the packets themself must possibly be rejected. Such an approach seems rather difficult and very likely to result in a faulty simulator behavior, due to the many low-layer adjustments that would be needed.

Neither for ns-3 nor for real-world instances it is intuitively clear where topology control should be mounted in the stack of protocols. So far there is no clearly favored opinion on this problem in the literature. Two protocols are most likely to be destined for the task: The routing protocol and the protocol used in the MAC-layer.

The routing protocol is used to decide to which node a packet must be sent, if a specific destination should be reached. For this task, it must have a list of neighbors that can be

[2]This does not happen very often, but it happens. This randomization is not according to RFC 2988 "Computing TCP's Retransmission Timer" [PA00].

reached and uses this list to decide which neighbor should be used to reach the destination efficiently. For changing node positions—which are not considered in this thesis—the routing protocol may also trigger the execution of the topology control algorithm dynamically if many links are reported to be broken.

The MAC layer controls access to the wireless medium. Hence detailed information about each communication link is at hand. Also, for a changing, mobile environment, the MAC layer is the first to know when the signal strength of communication links has significantly changed and the topology control algorithm should be executed again to provide an updated topology.

Since we do not consider changing node positions, restricting the routing protocol to neighbors that are selected by the topology control algorithm is sufficient. Our approach to restrict the routing protocols implemented in ns-3 consists of two steps:

1. Initially provide the routing protocol with a table of allowed communication links.
2. Discard all routing protocol control messages that are received from a neighbor that is not in the table of allowed communication links.

Since routing protocol messages are only considered from allowed communication links, it seems to the routing protocol as if messages to links that are not allowed have been lost (which is true, since they are dropped during reception at the routing layer of the receiving node). Hence, such not-allowed communication links are not added to the routing table. Packets for a node are routed through one of the allowed communication links to reach their destination. The corresponding modifications to the routing algorithms are described in Appendix A.1.

The OLSR and the DSDV routing algorithm both need some time to find the routes in the network as they are proactive routing algorithms (cf. Section 3.4). Hence we give the algorithms 30 seconds before the transmissions start in our experiments.

5.3.3. Test Instances and Testing Environment

To ensure comparability between the different topologies, we created fixed test instances determining the node positions and the sender-receiver pairs. We used the following parameters:

Number of nodes: For our simulations, the number of nodes varies between 40 and 60 nodes.

Area and node placement: On a square with base length of 50 to 550 meter, a (constant) position for each node is chosen randomly.

Number of sender-receiver pairs: Between 12 and 18 sender-receiver pairs—using each node only once—are chosen and saved along with the node positions in the instance file. We use a ratio of 3 sender-receiver pairs per 10 nodes. The amount of data to be transferred between each pair is 5 MB.

For each combination of the considered parameters, 25 instances are created. The exact parameters considered are given along with the experiments.

Testing environment: The experiments were run on the same machines as those given in Section 4.4.4. For randomization, the ns-3 seed is set to 1 for all experiments in this chapter. Each test instance is associated with its own run number. More information on the randomization in ns-3 is given in Section 4.4.4.

To create the test instances, python has been used. Python uses the Mersenne Twister as random number generator and is seeded by os.urandom.

Each experiment is repeated 25 times with different test instances. To ensure comparability between the topologies during an experiment, for each topology the same 25 input instances are used.

We have different representations for the results of the experiments. We mostly use

the median of our results along with error-bars which indicate the quartiles. Only in Section 5.4.2, the median of the results is sometimes given without quartile markers to enhance readability.

As some test instances (especially those that use relatively weak connection links combined with static routing) are not able to finish the transmissions, we used a threshold of 5000 seconds after which we assume the transmissions to be failed.

5.3.4. A Note on the Throughput

We will see in the next section that the time needed to finish transfers is surprisingly high for the considered topologies. It is well known that the theoretical bounds of 54 Mbit/s for wireless network using IEEE 802.11a (as well as for other standards) are not achievable in practice. For packet sizes of 512 byte, about 15-20 Mbit/s are reported to be possible [DANB00]. Using the ns-3 simulator and a distance of 50 meters between two nodes (which reduces the throughput), we achieved a maximum of 10 Mbit/s using the UDP protocol and about 8 Mbit/s using the TCP protocol. However, the results of our experiments imply that the throughput in the networks is considerably lower. For only one sender-receiver pair in the network, about 15 seconds are needed to transmit 5 MB, which equals roughly 350 kB/s or 2.5 Mbit/s.

Due to the use of topologies, the number of hops on a hop-minimal path between sender and reeiver is often relatively high even for communication partners that may be within transmission range of each other. To determine whether these hops lead to the lower throughput of the network, we conducted the following experiment:

For the experiment that lead to Figure 5.5, 7 nodes were placed in one row, with a distance of 50m between consecutive nodes (i.e., the x coordinate is 0 for all nodes and the y coordinate is 0, 50, 100, etc.). The used topology connects each pair of adjacent nodes, which means that all edges have length 50m. In this experiment, one sender-receiver pair had to transmit 5 MB of data over a varying number of hops. We measured the time needed to finish the transmission and calculated the throughput according to the time needed to transmit the data from the sender to the receiver.

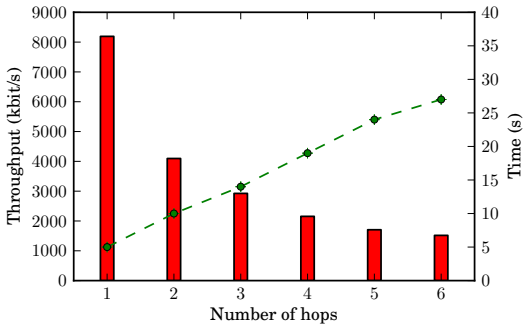


Figure 5.5.: Throughput vs. number of hops in a network consisting of a chain of nodes with 50 meters between two neighboring nodes. The bars show the throughput in kbit/s, while the dashed line describes the time needed to finish the transmission.

We can see that the throughput decreases rapidly as the number of hops is increased. This is due to the fact that most nodes are in the interference range of others. In this small setting, only one or two nodes can send at the same time, due to the CSMA/CA mechanism that delays transmission if another node within range is transmitting.

5.4. Experiments I

Comparing topologies is a difficult task, since there are many different quality criteria for topologies, as described in Section 5.1. Even worse, some of the criteria conflict with others. Low stretch factors, for example, often imply denser topologies as more communication links must be selected to achieve the low stretch factor. For other criteria, such as throughput or interference, results are based on mathematical models which abstract from aspects concerning the signal transmissions and protocol behavior, hence the impact on real applications is not clear.

In this section, we use the network simulator ns-3 to compare the considered topologies. By comparing the topologies regarding the throughput that can be achieved, we are able to study the influence of interference as well as other factors. This enables us to examine the influence of sparse topologies on the throughput-performance and ultimately the connection between sparse topologies and interference minimization. We consider two main scenarios for wireless sensor networks.

In the first scenario, which is considered in the remainder of this section, the topology control algorithm computes a topology that is used by the routing algorithm to direct traffic from senders to receivers. The senders and receivers may be separated by several hops, hence multi-hop communication may be needed. We use the IEEE 802.11a protocol with the parameters and modifications as described in Section 5.3. In the experiments, it is studied how long it takes to transmit a uniform amount of data from the senders to the receivers using the different topologies.

In Sections 5.4.2 to 5.4.4, we consider several parameters, such as restrictions on the link strength, varying amounts of workload and different node densities. The throughput-performance as well as the overall energy consumption are studied for fixed as well as for variable transmission powers in Section 5.4.5.

The second scenario uses the links of the considered topologies as input for scheduling algorithms. We use the scheduling algorithms that have been introduced in Chapter 4 to compare the performance of the topologies in combination with TDMA scheduling. In the literature, scheduling of all communication links in a topology is sometimes seen as a metric for interference induced by the topology (cf. [WW07, page 109-112]). However, we use a slightly different approach, which is introduced in Section 5.5, since we find that scheduling of all communication links in the topology prefers sparse topologies excessively. We consider the combination of scheduling and topology control in more detail in Section 5.5 along with experiments in Section 5.6.

5.4.1. Hop, Distance and Energy Spanner

Being a hop, distance or energy spanner is considered a valuable attribute for network topologies. If a topology control algorithm computes, for example, a 2-distance spanner, it is guaranteed that the signal must travel at most twice the optimal distance if only the communication links of the topology are used. Unlike most other algorithms that compute spanning topologies, for the spanner algorithms described in Section 5.2.8, $c \geq 1$ is a parameter that the algorithm uses to construct a c spanner. This enables us to compute spanner topologies for various values of c and hence to study the spanners for different stretch factors.

The parameter c is twofold. For a parameter c that is close to 1, only few links can be discarded. For a large c, however, a long detour, and therefore higher energy consumption, is probable. In order to decide for which parameter c good results are obtained, the following experiments were conducted. In these experiments, the time needed to finish all transmissions is measured.

We used an area of 200×200 meters is used and 60 nodes are randomly placed on this area for each test instance. 18 random sender-receiver pairs are associated to each instance. There is no restriction on the link strength in these experiments, but the algorithm considers strong communication links before weaker ones are considered. As described in

Section 5.3.3, for the different parameters c of the spanner algorithm the same test instances are used for better comparability. For hop, distance and energy spanners the time needed to transmit 5 MB of data between the 18 random sender-receiver pairs is measured. The main marker shows the median while the error-bars indicate the quartiles.

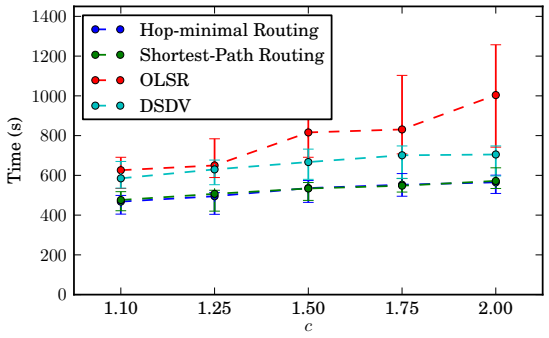


Figure 5.6.: Time needed to transmit all data for different c-distance spanners in dependence of c.

We consider the distance spanners first. As depicted in Figure 5.6, for values of c between 1.1 and 2, the time needed to finish all transmissions decreases as the parameter c decreases for all routing algorithms. The gain from having shorter distances and hence also usually less hops outweighs the disadvantage of the weaker links that are added as the distance stretch factor decreases.

We can also see that the static routing algorithms yield slightly better results for distance spanners than the build-in routing algorithms OLSR and DSDV. This is probably due to the lower overhead of the static routing algorithm as they do not update the topology information during the transmissions.

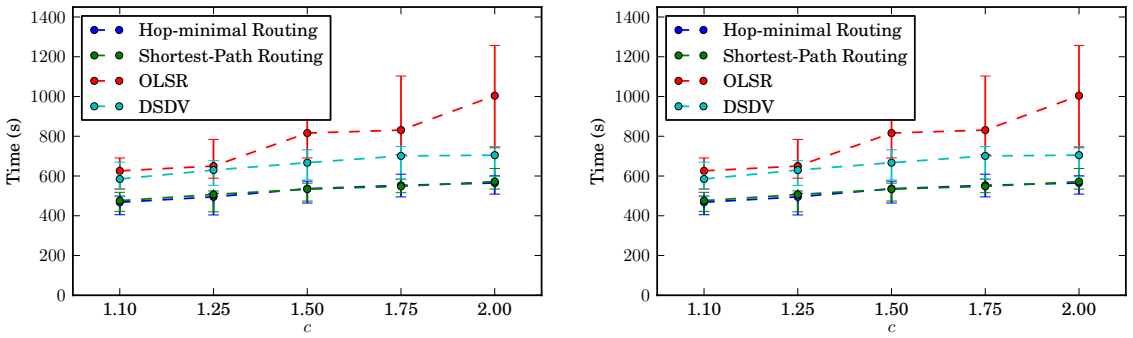
As the distance spanner yields the best results for the parameter $c = 1.1$, we will use the 1.1-distance spanner in the following experiments.

Regarding hop spanners, we can see in Figure 5.7(a) that the static routing algorithms, namely the hop-minimal and the shortest-path routing, can not achieve steady results for hop spanners. The hop-minimal routing algorithm does partially work for parameters between 4 and 6, but fails to finish the transfers for lower values of c. The shortest-path routing does not perform too bad, as the median of the time needed to finish the transfers for the shortest-path routing is generally the best value. However, this is highly dependent on the instance as for most parameters more than 25 percent of the instances have not finished the transfers within 1400 seconds.

This is due to the fact that for most instances some communication links with very low signal strength are selected since they span over large parts of the network. Those links are likely to be chosen by the shortest-path and the hop-minimal routing algorithms for some instances.

In Figure 5.7(b), the results are depicted for energy spanners with an energy stretch factor between 1.1 and 2. For the energy spanners, the static routing algorithms perform well. Both the hop-minimal and the shortest-path routing perform best for all energy stretch factors. This is due to the optimal routes as well as the lower overhead for the static routing algorithms.

For the OLSR routing we can see that the time needed to finish all transmissions increases more than for the other routing algorithms as the energy stretch factor increases. It can be observed that the OLSR routing is less performant regarding the throughput as the topology is getting increasingly sparse. This increase is probably connected to inefficient behavior of OLSR in sparse networks. The OLSR routing protocol has been designed



(a) c-hop spanners

(b) c-energy spanner

Figure 5.7.: Hop and energy spanner for different values of c.

for dense networks and it has been reported to have issues with sparse networks [FR09]. Those issues may be connected to the multi-point relays, which do not work properly in sparse networks, or to flooding of OLSR *TopologyControl* messages (cf. Section 3.4) in sparse networks. The DSDV algorithm shows a performance similar to the static routing algorithms but needs about 25 percent more time to finish the transmissions. The general overhead of the build-in routing algorithms in contrast to the hop-minimal and the shortest-path routing is probably again due to the higher overhead to maintain the routes and the fact that the computed routes are not necessarily optimal.

5.4.2. Restricting the Link Strength

Wireless reception hardware restricts communication to communication links that achieve a received signal strength higher than a certain threshold. If the energy of a signal is below this threshold, the signal can not be decoded—even without interference—due to the limited sensitivity of the receiver. In ns-3, this energy is defined by the energy detection threshold of -96 dBm.

However, a topology control algorithm can also restrict the link strength by selecting only communication links that exceed a certain threshold. A simple algorithm that uses such a threshold is the RLS algorithm described in Section 5.2.7. RLS has a similar behavior as the spanner graphs: For a low threshold, only few edges are removed. For a high threshold, the throughput in the resulting network topology may suffer, as only few links are available or the network may not even be connected. Therefore, a good threshold has to be found.

In this section, we study how the throughput of the network is affected by restricting the topology to links whose signal strength exceed a certain threshold. For the experiments, a setup of 60 nodes on a square with 200 meter base length is used. As usual, each one of 18 sender-receiver pairs had to transfer 5 MB across the network.

Before we determine a good threshold, we first study how the restriction on communication links with high signal strength influences the connectivity of the network. Since the RLS algorithm does not necessarily compute connected topologies, this must also be considered for an optimal threshold. In Figure 5.8, the percentage of our instances that are not connected if the signal strength is restricted to the according threshold is displayed. Note that for the results in the following experiments in this section, only those instances are considered that are connected and hence the number of considered instances decreases for restrictions to communication links with high signal strength.

In order to determine a signal strength threshold which allows good throughput, we measure the time needed to finish the transmission between the 18 random sender-receiver pairs for various thresholds. We use the same setup as in the previous experiment in this

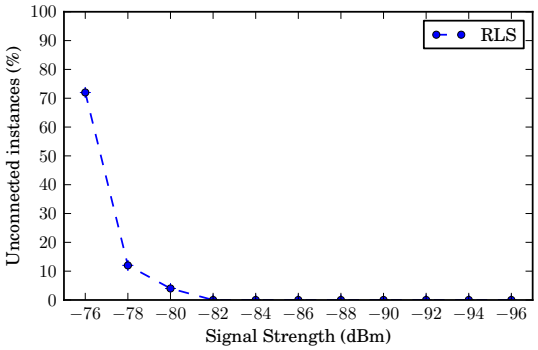


Figure 5.8.: Percentage of unconnected instances for the RLS algorithm using different thresholds for the restriction of links.

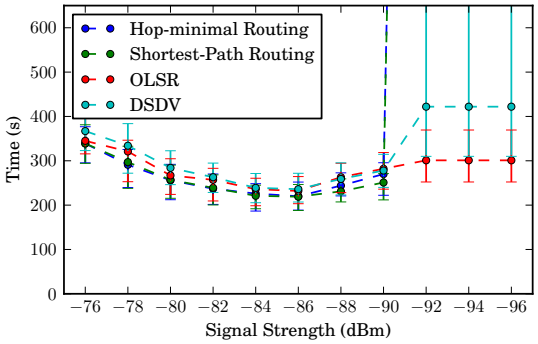


Figure 5.9.: Time to finish all transmissions for restricted link strength topologies using the four different routing algorithms. We can see that the required time is lowest for a threshold of -84 dBm or -86 dBm.

section. Note that the number of instances decreases as we restrict the communication links to those that exceed a high threshold according to Figure 5.8.

In Figure 5.9, we can see that if the RLS algorithm is limited to only few but very strong links, the resulting topology achieves acceptable, although not optimal throughput-performance. The results are not optimal mainly because the stronger communication links are shorter and hence more hops are needed to reach the receiver. However, we have seen in Figure 5.8 that many instances are not connected anymore for restrictions to higher signal strength than -80 dBm.

If many links are used by the algorithm, including some with low signal strength, routing on the topology does not perform well either. For the static routing algorithms, this is due to the fact that the routes are chosen prior to the transmission and communication links with low signal strength can not be avoided. Both, OLSR and DSDV keep track of communication links that appear to be broken and avoid them. Hence, those links that result in high losses are not used. Thus, for OLSR and DSDV, the time needed to finish the transmissions does not increase as dramatically as for the static routing algorithms. However, the routing algorithms still select some communication links with low signal strength and since the rate-control manager automatically decreases the data-rate for links with low SINR, the performance of those routes is not optimal. Overall, the OLSR routing protocol seems to be more robust against links with low signal strengths than the DSDV routing algorithm.

We can observe that for thresholds of -86 dBm and -84 dBm, the throughput-performance of the routing algorithms is best on the resulting topologies.

Considering the throughput-performance as well as the expected connectivity of the instances, we come to the conclusion that the RLS topology control algorithms with a threshold of -86 dBm yields the best results. Hence, we will use the RLS algorithm with this threshold for our experiments; the RLS algorithm using this threshold is denoted by $\text{RLS}_{\text{-86 dBm}}$.

The link strength can also be restricted for other topology control algorithms. This may result in different outputs and influence the properties of the generated topologies. Also, by comparing the performance of the unrestricted topologies with the performance of the restricted topologies, one may be able to draw conclusions about the signal strength of the links that are mainly used in the topologies.

Usually the topology control algorithms can select a subset of the set of all possible connection links. In this experiments, however, we restrict the input of the topology control algorithms to the connection links whose signal strength exceeds a certain threshold. The topology control algorithms must compute their topology solely based of this subset of links.

To study the performance of the topologies, the time they need to finish the transmission between the sender-receiver pairs is measured. In Figure 5.10, this time is depicted for various thresholds on the restriction on the input for the topology control algorithms. For the experiments that led to those results, we used a setup of 60 nodes, randomly placed on a square area with base length of 200 meters. Each of the 18 random sender-receiver pairs had to transmit 5 MB of data. We used 25 instances for this experiment; In the depicted results, the main marker indicates the median of the time needed to finish the transmissions and the error-bars show the quartiles.

For the EMST topology used in Figure 5.10(b), the OLSR routing algorithm does not perform very well. The additional time needed to finish the transmissions is primarily due to the overhead created by the messages sent by the OLSR algorithm along this very few edges. Additionally, as described for the energy spanners, the OLSR algorithm does not perform well for very sparse networks.

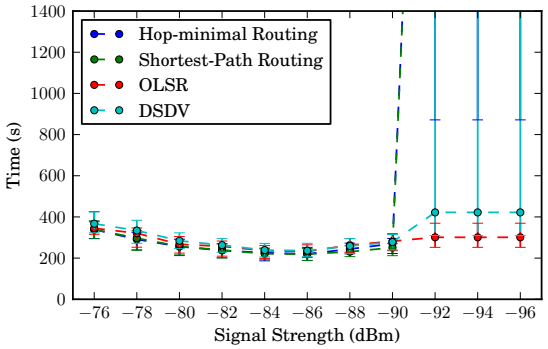
For thresholds below -84 dBm the time needed to transmit all data for the considered instances decreases slightly. This is propably since the topologies that required relatively long links in the minimal spanning tree are not considered anymore, since they are not connected anymore (cf. Figure 5.8). For all routing algorithms except the OLSR routing, the results achieved by the EMST topology is relatively constant.

The results of the XTC algorithm are depicted in Figure 5.10(c). We can see that the performance of the topology computed by the XTC algorithm does not change very much when the subset is restricted to communication links with high signal strength. Since XTC is based on a neighborhood criterion, mostly short links are chosen and hence the restriction on links with high signal strength has only marginal impact on the performance of the topology computed by the XTC algorithm.

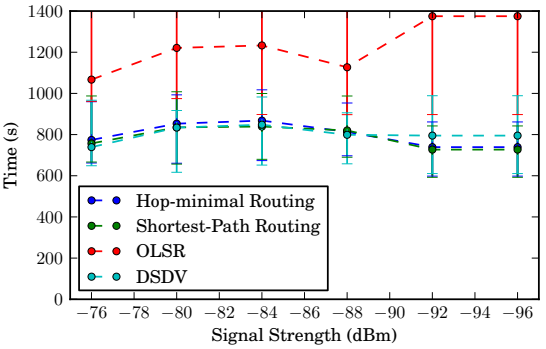
The topology based on the Yao graph, whose results are depicted in Figure 5.10(d), achieves similar results as the XTC algorithm. However the throughput is slightly better and the difference between the routing algorithms is lower. Similar to the XTC topology, the performance does only decrease slightly as relatively strong links are discarded.

The time needed to finish the transmissions for the topology based on the Gabriel graph is depicted in Figure 5.10(e). The GG topology shows similar performance as the XTC and the YG_6 topologies. However the throughput-performance of the GG-based topology is in-between the two previously mentioned topologies.

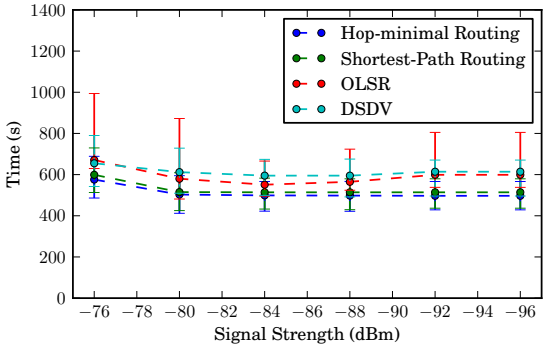
The results for the different spanner topologies, computed according to the algorithm described in Section 5.2.8, are depicted in figures 5.10(f) to 5.10(h). We can see that the 2-hop spanner depicted in Figure 5.10(f) yields visually similar results as the RLS graph, however, it is about 25 percent slower. For the build-in routing algorithms the performance is best between -84 dBm and -88 dBm while the topology does not achieve stable results



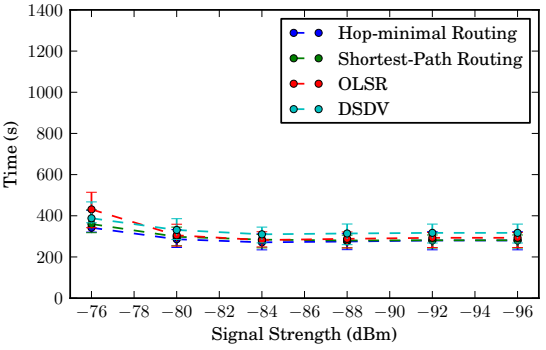
(a) RLS



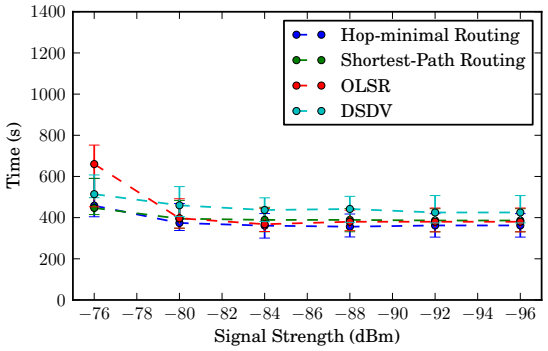
(b) EMST



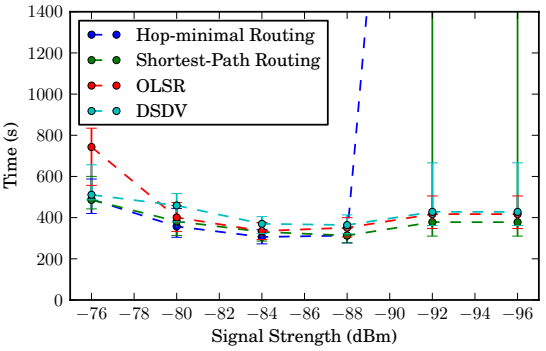
(c) XTC



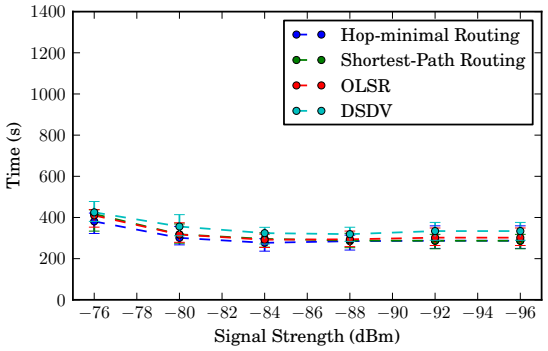
(d) YG_6



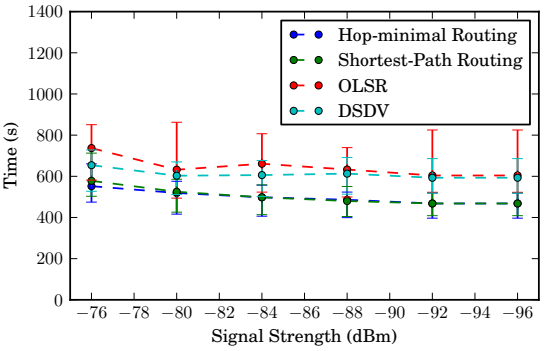
(e) GG



(f) 2-hop spanner



(g) 1.1-distance spanner



(h) 1.1-energy spanner

Figure 5.10.: Comparing topologies for different restrictions on the set of possible communication links. This set is restricted to the links whose signal strength exceeds the considered threshold. Note that the number of instances decreases for higher threshold values as some instances are not connected anymore (cf. Figure 5.8).

for the static routing algorithms if all links are allowed. This is again due to the long, relatively weak links that may be added and that are likely to be chosen especially from the hop-minimal routing algorithm. The 1.1-distance spanner achieves stable results that are very similar to the YG_6. For the 1.1-energy spanner we can see a moderate increase in the time needed to finish the transmissions for a restriction on strong communication links. Since mostly short and hence energy-efficient links are used, this is not surprising. Both built-in routing protocols show a slightly worse performance than the static routing. Again, this is probably due to the overhead which is needed to keep the routes updated.

On an overall comparison of Figure 5.10, we see that the $RLS_{\text{-86 dBm}}$ topology, which uses all available communication links with a signal strength higher than -86 dBm, results in slightly higher throughput than the YG_6 and the 1.1-distance spanner. The EMST algorithm does not yield a good performance, especially for the OLSR routing algorithm. It generally seems as if the built-in routing algorithms create a slightly higher overhead for sparse topologies. Without restricting the link strengths, the ALG topology does not yield acceptable results, and hence another topology should be used if relatively weak links are necessary to ensure connectivity.

Overall we can see that for most topologies a combination with a restriction on stronger links is not necessary. In the ALG, if all edges are used, a restriction yields considerably better results. Similarly, the 2-hop spanner yields better results if combined, the differences however are not as significant as for the ALG.

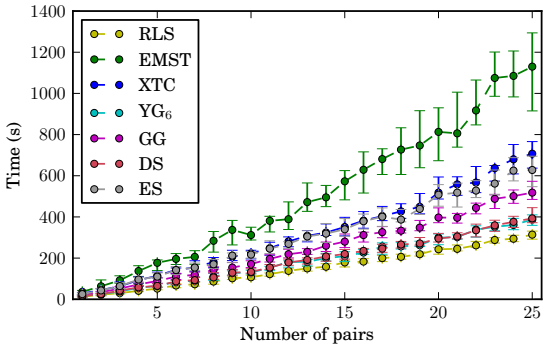
5.4.3. Increasing the Workload

In order to see how the different topologies can cope with a varying amount of traffic, we slowly increase the number of sender-receiver pairs in the network in the experiments considered in this section.

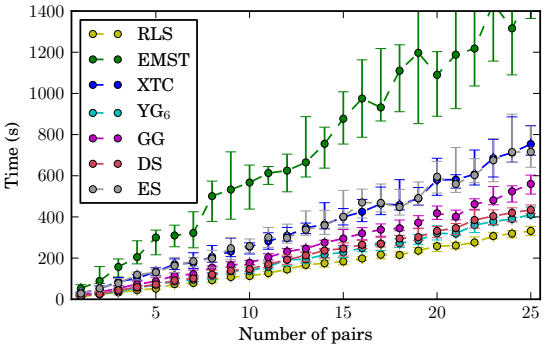
The experiments conducted in this section use 50 nodes. The nodes were randomly positioned on squares with base lengths of 200 meters and 400 meters, respectively. For each number of pairs, 25 random instances are considered. The set of communication links is generally not restricted. Only for the $RLS_{\text{-86 dBm}}$ the signal strength is set to -86 dBm and for the EMST it is set to -90 dB to enable relatively good results even for the OLSR routing algorithm. Since the hop-minimal routing showed similar results as the shortest-path routing and the combination of parameters is relatively high for these experiments, we conducted these experiments only for the shortest-path, the OLSR and the DSDV routing. In the following experiments, we measured the time the topologies needed to transmit the data from the senders to the receivers.

In Figure 5.11, we can see that the time needed to finish the transmissions is similar throughout the different routing algorithms in the 200×200 meter setting. Only the EMST performs worse for the OLSR routing algorithm, as the OLSR routing does not perform well on sparse topologies. The topology that performs best for almost all numbers of sender-receiver pairs is the $RLS_{\text{-86 dBm}}$ topology, which uses all communication links whose signal strengths exceed the threshold of -86 dBm. Since the link strength is restricted to sufficiently good links, this topology enables high throughput while offering various routes and hence avoiding bottlenecks. The topologies seem to be ordered roughly according to the densities which can be seen in the visual comparison of the topologies in Figure 5.4. The best performance besides the $RLS_{\text{-86 dBm}}$ topology is achieved by the YG_6 and the 1.1-distance spanner, followed by the topologies based on the Gabriel graph, the 1.1-energy spanner and XTC.

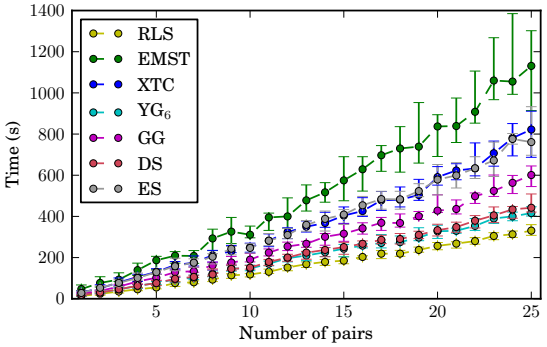
To confirm the assumption that the throughput performance relies on the node density of the network, as well as to study the different topologies more intensively, we now examine some basic properties of the considered topologies. For the average number of hops in Figure 5.12(a), the hop-minimal path from each sender to the corresponding receiver has been calculated and the mean of the number of hops is stored. The average edge degree, which is shown in Figure 5.12(b), is calculated based on the links chosen by the topologies.



(a) Shortest-path routing

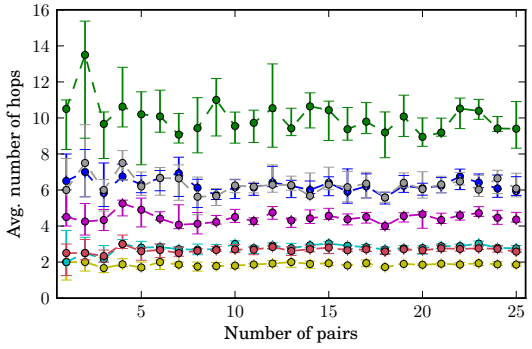


(b) OLSR routing

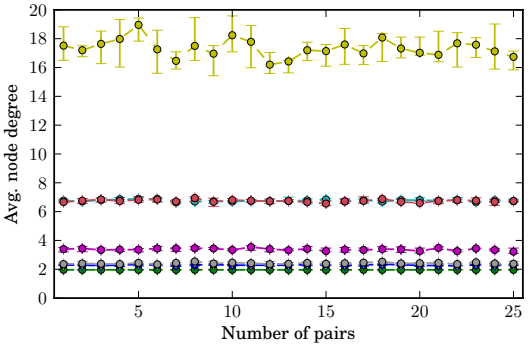


(c) DSDV routing

Figure 5.11.: Comparing topologies using an increasing number of random sender-receiver pairs that need to transmit data for the different routing algorithms. The used area is a square of 200×200 meters.



(a) Average number of hops



(b) Average node degree

Figure 5.12.: Basic properties of the topologies considered in this experiment. For a legend, cf. Figure 5.11.

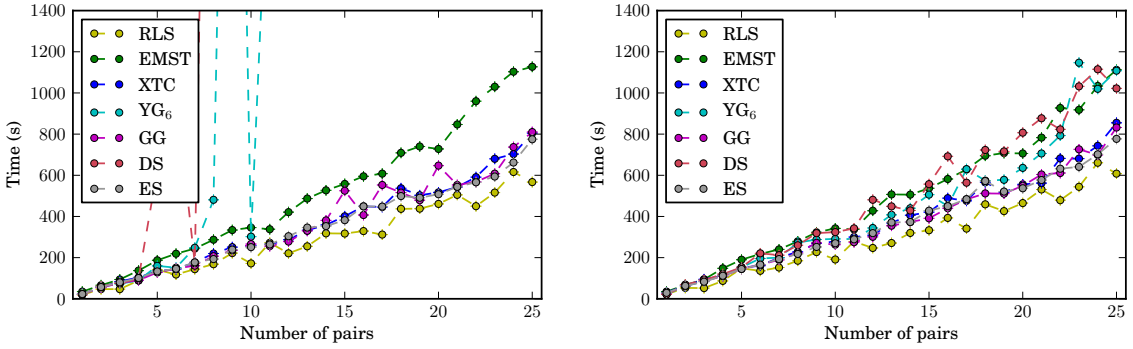
The average number of hops is not accountable for the increase in time needed to finish the transmissions as the number of hops increases. However, by comparing Figure 5.12(a) and Figure 5.11, we can see that the time needed to finish the transmissions (regardless of the number of transmission pairs considered) and the average number of hops are closely related. The order of the topologies are the same for both measures and even the relative

differences between the topologies are similar for the time needed to finish the transmissions and the average number of hops.

Regarding the average node degree of the topologies, which is depicted in Figure 5.12(b), the values extracted from the topologies confirm what we have seen in the visual comparison in Figure 5.4. The average node degree is very low and between 2 and 4 for the EMST, XTC and the 1.1-energy spanner. For the 1.1-distance spanner and the YG_6, the average node degree is about 7 and the $RLS_{\text{-86 dBm}}$ has an average node degree of roughly 15.

We may note that mainly for those topologies that need more time to finish the transmissions there is a high fluctuation in the average time to process all transmissions. This can be seen for the topologies based on XTC and the 1.1-energy spanner and especially for the EMST. For those topologies, only few links connect various sender-receiver pairs. Especially for the EMST, there is only one path between each sender and receiver. For this reason, there are probably many communication links that are used by several sender-receiver pairs. Those links are very likely responsible for several failed TCP connections and hence delayed transmission.

There are many properties that differ between sparse and dense topologies, e.g., the average node degree and the average edge length. However, the number of nodes per square-meter gives another measure of density, which does not depend on the used topology. In our next experiments, we consider a similar setup as earlier in this section on a larger area. The number of nodes remains at 60 nodes, but they are placed on an increased area of 400×400 meters. Hence, the node density is 4 times lower (in nodes per square-meter) than in the previous setup. The results of these experiments are depicted in Figure 5.13 and Figure 5.14. Not that the results in Figure 5.13 show only the median, an according figure that depicts the median along with the quartiles can be found in Appendix A.3.



(a) Shortest-path routing

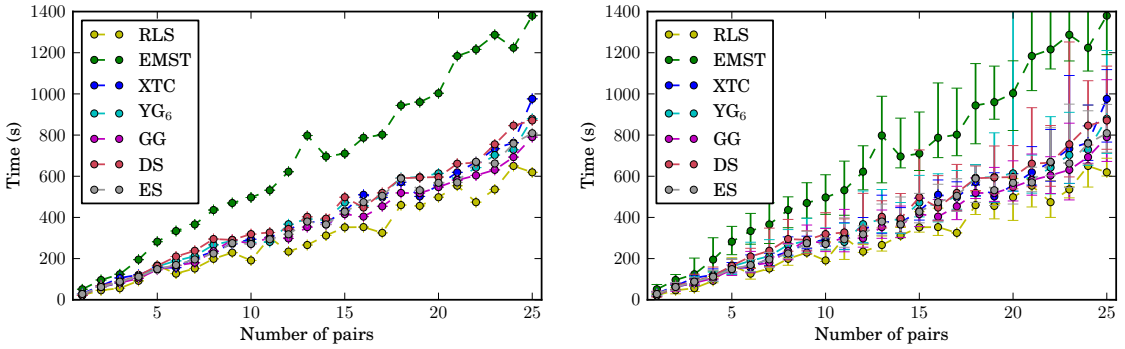
(b) DSDV routing

Figure 5.13.: The time needed to finish the transmissions for the relatively sparse topologies on an area of 400 × 400 meters. The same figures with quartile markers can be found in Appendix A.3.

For the shortest-path routing algorithm, we can see in Figure 5.13(a) that the topologies based on the 1.1-distance spanner and the Yao graph do not produce good results. Both topologies are relatively dense and hence select many communication links. Due to the increased deployment area, the topologies are selecting communication links that are too weak to enable communication with a good performance. Due to interference some communication links may fail completely. Since those weak links can not be avoided using the static routing protocol, some transmissions can not be finished. The densest topology, however, is probably the RLS algorithm which still yields very good results. This is due to its restriction on relatively strong links. For the other topologies, namely the 1.1-energy spanner, the XTC, the Gabriel graph and the EMST their performance is again

roughly according to the average number of hops that are used (cf. Figure 5.15(a)). Only the Gabriel graph is slightly worse than expected if only the average number of hops is considered. This is probably due to longer and hence weaker communication links (cf. Figure 5.16(b)).

The results for the built-in routing algorithm DSDV are depicted in Figure 5.13(b). We can see that the resulst for those topologies that have been able to finish for the shortest-path routing algorithm is mainly similar. However, due to the possibility to avoid weak links in this routing algorithm, the 1.1-distance spanner and the Yao Graph can yield better results. But still the results are considerably worse than for the denser setup, especially if the results are considered relative to those of the other topologies.

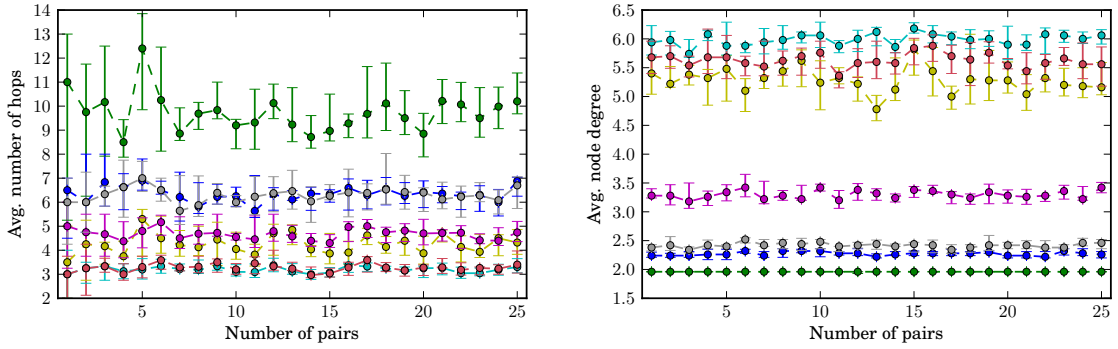


(a) OLSR routing without quartile markers

(b) OLSR routing with quartile markers

Figure 5.14.: The time needed to finish the transfers for the OLSR routing algorithm. This is the only routing algorithm that achieves relatively stable results for almost all considered topologies on an increased area of 400×400 meters.

For the OLSR algorithm, whose results are depicted in Figure 5.14, most topologies achieve relatively similar performance. This is since the OLSR routing algorithm is relatively robust against weak links. The RLS topology is slightly better than the other topologies, which is due to its combination of relative few hops with communication links that are strong enough to allow sufficient throughput. As expected, the EMST topology is worse for the OLSR algorithm. This is due to OLSRs worse performance on sparse topologies (cf. Section 5.4.1).



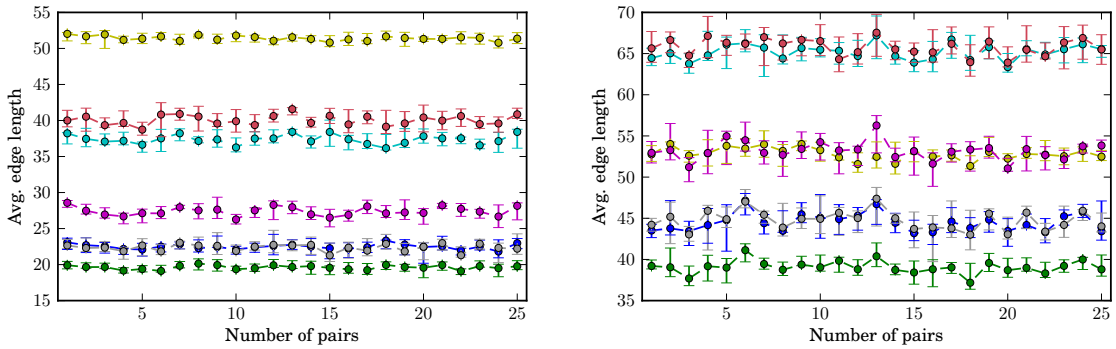
(a) Average number of hops

(b) Average node degree

Figure 5.15.: Properties of the considered topologies for the setup on the area of 400×400 meters. For a legend, cf. Figure 5.14.

However, by comparing Figure 5.12 and Figure 5.15 we can see that the differences between the setup on 200 × 200 meters and 400 × 400 meters are not due to an increased number of hops or a differing edge degree as these values are similar for both setups. Only the values for the RLS$_{\text{-86 dBm}}$ topology differ significantly. For the denser setup, about 2 hops are needed on average while on the less dense setup 4 to 5 hops are needed. This is since the edge length is limited relatively strictly to about 50 meters for the RLS$_{\text{-86 dBm}}$.The average node degree decreases from about 15 in the dense setup to about 5 in the less dense setup. This is due to relatively short edges and the increased distance between the nodes.

One of the properties that clearly differs for the different setups is the edge length as depicted in Figure 5.16. We can see that the average edge length increases by about 20 meters for each topology. Only for the topology based on the RLS$_{\text{-86 dBm}}$, the average edge length does not increase. Again, this is due to the restricted signal strength and hence a lower maximal edge length.



(a) Average edge length for 200 × 200 meters

(b) Average edge length for 400 × 400 meters

Figure 5.16.: Average edge length for the considered topologies on an area of 200 × 200 and 400 × 400 meters. For a legend, cf. Figure 5.14.

5.4.4. Density

In the last section, we have seen that the density is important for the throughput performance of the different topologies. However, we did not finally settle the question which topology properties are mainly responsible for different performances for dense or not-so-dense topologies. Since the density of a network is an important parameter for the topology control algorithms, we will consider the influence of the density in more detail in this section. With the results given in this section, one can also choose a topology that fits the needs for the application at hand, based on the density that is expected for the considered application.

We compare the topology control algorithms using 60 nodes randomly placed on square deployment areas with base lengths of 50 to 600 meters. The topologies must process the transmission of 5 MB of data between each one of 18 random sender-receiver pairs. Most topology control algorithm received all possible communication links, without restrictions on the signal strength, as input. Only the RLS$_{\text{-86 dBm}}$uses only links whose signal strength exceeds -86 dBm and the EMST uses only links whose signal strength exceeds -90 dBm. We will first consider the percentage of unconnected instances for the varying deployment areas, in Figure 5.17. It can be seen that once the deployment area exceeds 500 × 500 meters, a considerable percentage of the instances are not connected anymore, even without restriction on the signal strength of the communication links.

To compare the throughput performance of the topologies for varying deployment areas and hence varying densities, we measure the time the topologies need to process transmissions between 18 sender-receiver pairs. The results of these experiments are depicted in

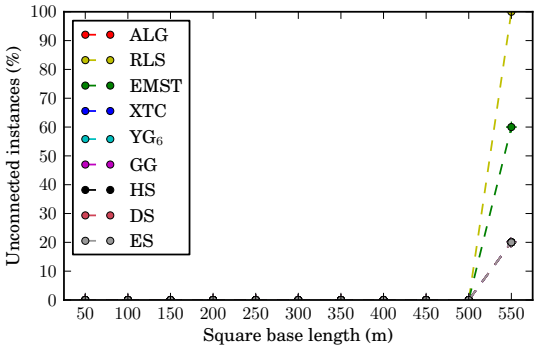


Figure 5.17.: Connectivity of the topologies for the different sizes of the deployment area.

Figure 5.18. Once a density is reached, for which the median of the time needed to finish the transfers is higher than 2500 seconds, or if all instances of the topology are unconnected the results for this topology are no longer displayed. Note that the number of used instances decreases for some topologies differently than for others. For more information compare Figure 5.17.

From the two static routing algorithms, only the shortest-path routing is depicted. The results for the hop-minimal routing are similar due to the similarity of the routing algorithms. First some basic observations for the shortest-path routing algorithm. The topology without restrictions, ALG, does not perform well. Once a deployment area larger than 100×100 meters is considered, the transmissions can not finish anymore. This is due to weak links, which are not avoided by the static shortest-path routing algorithm. However, for up to 100×100 the ALG yields good results since it uses mostly one-hop connections.

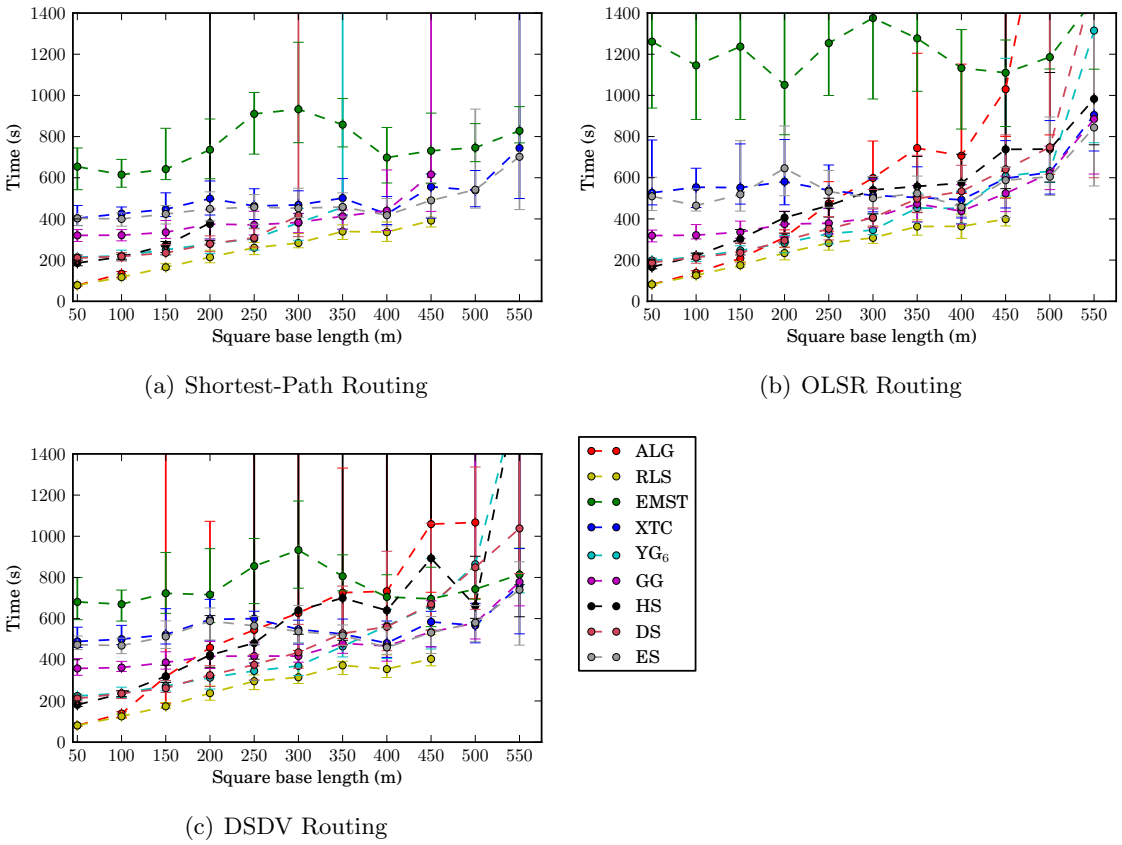
For the static shortest-path routing, it can also be observed that the 2-hop spanner, the 1.1-distance spanner, the Yao Graph as well as the Gabriel graph are not able to finish the transfers anymore for larger deployment areas in the shortest-path routing. The majority of the transmissions can not be finished within 2500 seconds anymore for the 2-hop spanner once the base length of the deployment area exceeds 200 meters, similarly for the 1.1-distance spanner once 300 meters are exceeded, for the Yao Graph once 350 meters are exceeded and once 450 meters are exceeded for the Gabriel graph.

This is probably due to the increasing edge lengths for larger deployment areas. As depicted in Figure 5.19(a), for all those topologies the average edge length is about to exceed 50 meters when the topologies are no longer able to finish the transfers. However, this is probably not the exact reason why the topologies are no longer able to finish the transmissions. As more very long links are selected to be in the topology, the average edge length increases along with the probability that those links are chosen by the routing algorithm. If those links are too weak, a successful transmission is hardly possible.

The topology based on RLS achieves the shortest time to finish the transmissions for almost all densities. However, for deployment areas that exceed a base length of 450 meters, the topology is only seldom able to finish the transmissions

The three remaining topologies, namely those based on the EMST, XTC and the 1.1-energy spanner, are relatively sparse and use only very few, relatively strong connection links. This enables these algorithms to finish the transmissions for large deployment areas even though the distances between the nodes are relatively high and hence almost only weak communications links are available. Considering only those three topologies, the 1.1-energy spanner and the XTC algorithm yield the best results. The EMST needs at least for dense networks considerably more time to finish all transmissions.

Considering denser deployment area, those three topologies need more time to finish



(a) Shortest-Path Routing

(b) OLSR Routing

(c) DSDV Routing

Figure 5.18.: Comparing topologies for varying densities for the different routing algorithms. The density is varied by increasing the base length of the square into which the 60 nodes are randomly placed.

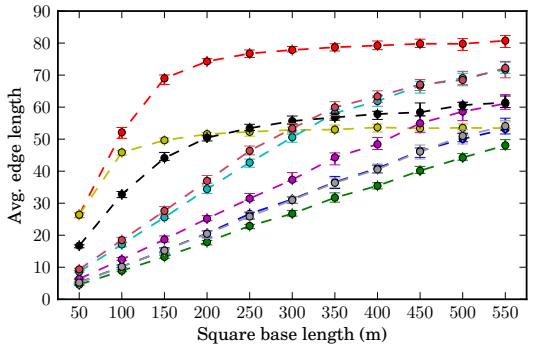
the transmissions than the other topologies, since they need more hops to route the traffic from the senders to the receivers (cf. Figure 5.19(b)).

For both the OLSR and DSDV algorithm, the $\text{RLS}_{\text{-86 dBm}}$ topology is superior to the other topologies for deployment areas up to 450 × 450 meters. For larger areas, the topologies based on XTC, the 1.1-energy spanner and the Gabriel graph achieve the best performance for both routing protocols.

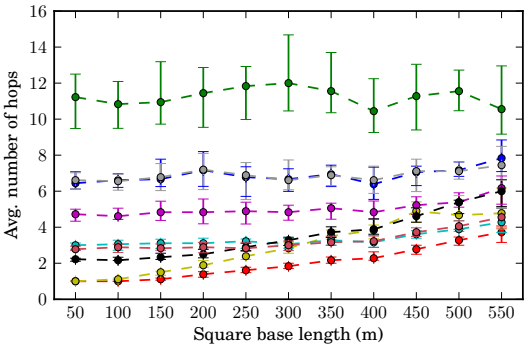
Overall, it can be said that since OLSR and DSDV are more robust against weak links, for more topologies (for example the 2-hop spanner or the ALG) the transmissions are finished. Especially the OLSR routing protocol is robust against weak links, while the DSDV routing protocol achieves a better performance for rather sparse topologies, especially the EMST.

The Yao graph and the 1.1-distance spanner achieve relatively similar results for the build-in routing algorithms. Their throughput is good for deployment areas with a base length of up to about 350 to 400 meters while, only the $\text{RLS}_{\text{-86 dBm}}$ topology achieves a better performance. For larger areas, the sparser topologies are superior.

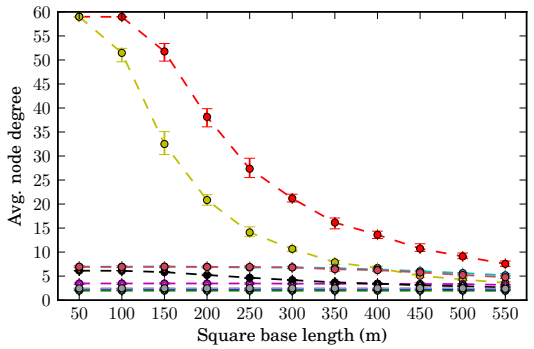
We have seen in the experiments in this section that the shortest time to finish the transfers and hence the best throughput can be achieved for not too long edges in combination with as few hops as possible. For too long edges, the data rate is restricted while for to many hops the throughput decreases considerably—due to interference and additional transmissions. We have seen that even though the built-in routing algorithms can avoid weak links, the throughput-performance can be improved by using a topology that discards those weak links completely.

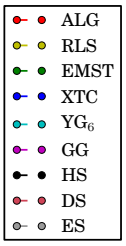


(a) Average edge length for various densities



(b) Average number of hops for various densities





(c) Average node degree for various densities

Figure 5.19.: Properties of the considered topologies depending on the considered deployment areas.

However, for networks with a very low density, long edges must probably be included in the topology. For such applications, the sparse topologies such as those based on the XTC, EMST or the 1.1-distance spanner may be considered. Due to their restriction on relatively short links regardless of the density of the network, the throughput for networks with high density decreases considerably.

5.4.5. Reducing the Transmission Power

Topology control aims mainly at computing a subgraph of all possible communication links that minimizes interference and energy consumption. As neither interference nor the energy consumption are minimized if only links that are allowed for communication are limited, so far we compared the topologies without exploiting a possible advantage. In the previous sections, we considered communication with a fixed, common transmission power. In this section, we consider scheduling with variable transmission powers for each node. Using fixed as well as variable transmission powers, we compare the throughput performance as well as the energy consumption of the topologies.

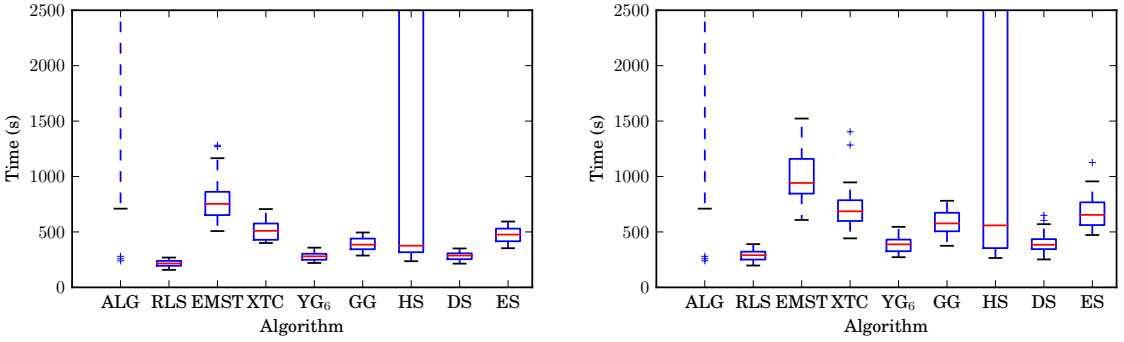
In order to achieve minimal transmission powers such that each communication link that is chosen can be used, it must be ensured that the minimal possible transmission power grants a sufficient signal strength for each communication link. The minimal possible transmission power for each node is computed such that the neighbor with the least reception signal strength that is chosen by the topology achieves a reception signal strength of -90 dBm. If this transmission power would be higher than the default value of 16.0206 dBm[3], this value is not increased. The threshold of -90 dBm has been chosen based on the results

[3] 16.0206 dBm is the default transmission power. A transmission gain of 1 dB is added by default and accounted for in our minimal transmission powers calculations.

in Section 5.4.2, as for topologies with a signal strength of -90 dBm or above the routing algorithms can usually finish the transmissions.

As reducing the transmission power does also reduce the range of the node's signal, less nodes are interfered by the signal. Hence, a faster transmission may be possible. It is also probable that an overall improvement of the energy consumption can be achieved.

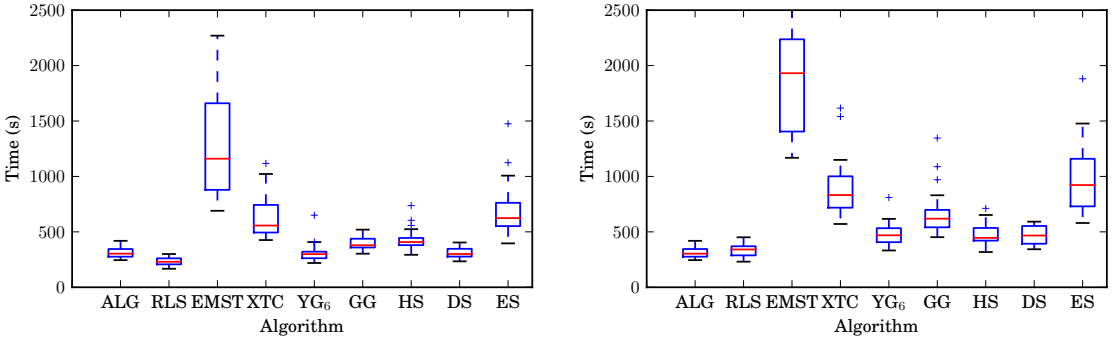
We will first consider the throughput performance. As before, we use the time needed to transfer the data between random sender-receiver pairs as a measure for throughput performance. Our experiments consists of 60 nodes that are randomly placed on a square area of 200×200 meters. 18 random sender-receiver pairs are required to transmit 5 MB of data. In Figure 5.20 and Figure 5.21 the results of these experiments are displayed.



(a) Fixed transmission powers

(b) Variable transmission powers

Figure 5.20.: Time needed to finish the transmissions using the shortest-path routing algorithm for fixed transmission powers and variable transmission powers.



(a) Fixed transmission powers

(b) Variable transmission powers

Figure 5.21.: A comparison of the time needed to finish the transmissions using the OLSR routing algorithm for fixed transmission powers and variable transmission powers.

For both routing algorithms, we can see that the time needed to finish the transfers increased and hence the throughput of the algorithms decreased if variable transmission powers are used. Solely for the ALG algorithm this is not the case, since a reduction of the transmission power is usually not possible as almost every node has neighbors far away with a signal strength even below -90 dBm.

There are several reasons why the throughput did not increase but decrease as the transmission power and hence the interference on other nodes is reduced. One reason is that the signal strength that is received at the intended receiver is also lower, which

implies that more messages may be lost due to a lower SINR. Another reason is that the SINR that can be achieved over a connection link determines the data rate that is used for the transmission over this link. As the signal strength is reduced, the SINR decreases accordingly and hence a lower data rate may be achieved.

Another reason may be the relatively small setup. As interference propagates far and does also affect nodes further away, the benefits from reducing the transmission powers may be more relevant for larger and more wide-spread networks.

So far, variable transmission powers did not show their benefits is our experiments. Considering the energy consumption, however, it is more likely that variable transmission powers are beneficial.

To study and compare the overall energy consumption, we need a measure thereof. As communication is consuming major parts of the energy, we assume that the energy consumed is proportional to the number of data packets sent. As not only data packets but also many smaller ACK and MAC layer packets are sent, we distinguish between data packets and maintenance packets. Those maintenance packets are usually about 5 to 40 times smaller than data packets. As data packets are 588 bytes (512 bytes payload plus headers), we assume that one data packets needs at least 1/12000th of a second to be transmitted[4]. We assume that an average maintenance packet needs about 4 microseconds to be transmitted. Hence, we calculate the energy in joule (J) that is consumed for one run according to

$$\text{consumed energy} = \text{ATP}_\text{W} \cdot \left(\frac{\#\text{ data packets}}{12000} + \frac{\#\text{ maintenance packets}}{240000}\right) \tag{5.1}$$

where ATP_W is the average transmission power of the nodes in the used topology in watt, "# data packets" is the number of all data packets sent during the transmission of the data and "# maintenance packets" is the number of maintenance packets that are sent until the transmissions have been finished.

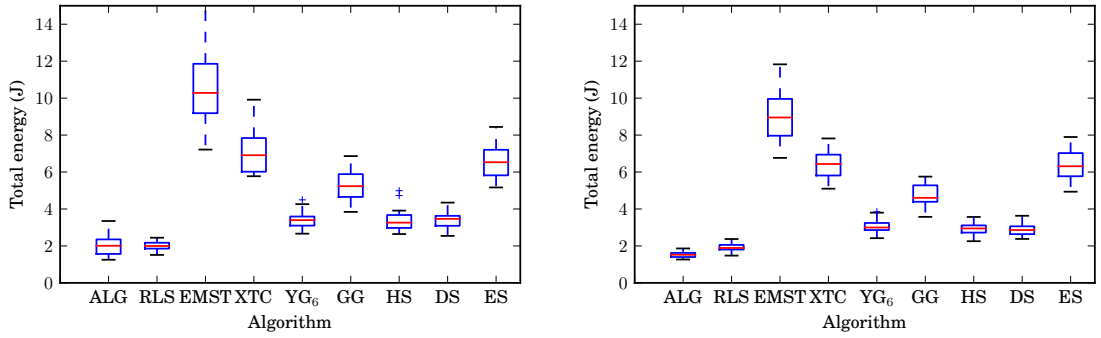
To calculate the overall consumed energy of the topologies during the transmission the average transmission power of each node is extracted and the mean over all nodes is calculated. During the simulation, each packet that is sent is categorized as data or maintenance packet and counted accordingly. Using these values, we can compute the overall consumed energy as defined above.

In Figure 5.22 and Figure 5.23 we examine the energy that is consumed during the transmission of the data for the different topologies using the shortest-path and the OLSR routing. In Figure 5.22 the energy consumption is depicted for fixed transmission powers while Figure 5.23 depicts the energy consumption for variable transmission powers. For the experiments, 60 nodes are placed randomly on a square area of 200 meter base length and each one of the 18 sender-receiver pairs must transmit 5 MB of data using the communication links chosen by the topology control algorithms.

In Figure 5.22, the total energy consumption of the topologies, computed according to Equation (5.1), is depicted for the shortest-path routing and OLSR routing. We can see that the consumed energy is closely related to the time the topologies needed to finish the transmissions. This is due to the fact that each sender tries to send its data as soon as the CSMA/CA mechanisms allows the node to send. Also, since the throughput decreases as the number of hops increases (cf. Section 5.3.4), those networks that use a large number of hops (and hence require many transmissions) yield a lower throughput performance.

The total energy consumed by the topologies with variable transmission powers is displayed in Figure 5.23. The transmission power is computed for each node individually as the lowest possible power such that for each communication link that is included in the topology the communication partner can be reached with a signal strength of at least -90 dBm.

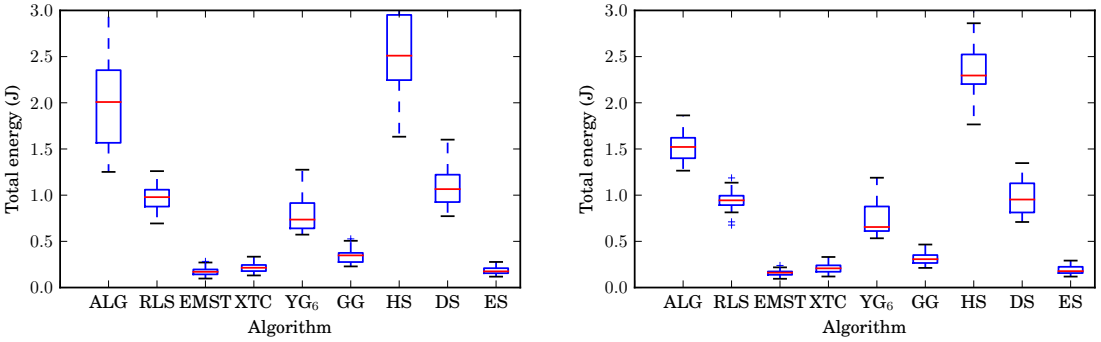
[4] 1/12000th of a second would be achieved by a 54 Mbit/s connection link.



(a) Shortest-path routing

(b) OLSR routing

Figure 5.22.: Energy consumption of the topologies without reduced transmission power. The total energy consumption is depicted for the shortest-path and the OLSR routing algorithms.



(a) Shortest-path routing

(b) OLSR routing

Figure 5.23.: The total energy consumption of the topologies with variable transmission powers for the shortest-path and the OLSR routing algorithms.

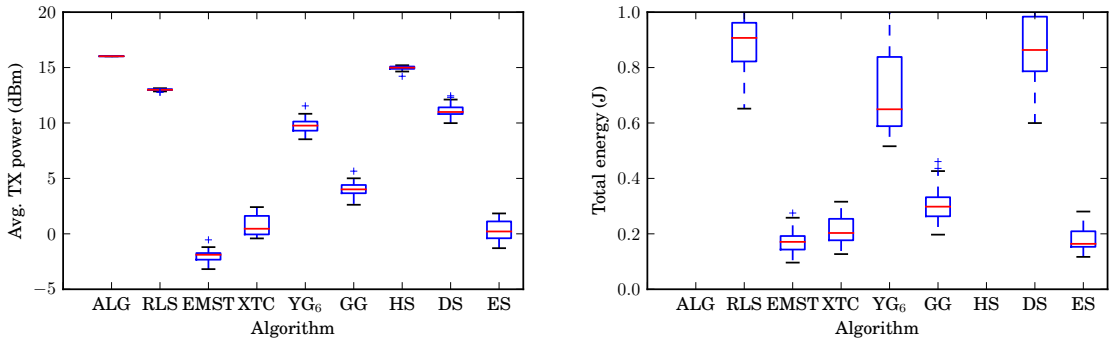
Our experiments show that, once variable transmission powers are enabled, the time needed to finish all transmissions no longer determines the energy consumption. Topologies such as the EMST, the 1.1-energy spanner, the XTC algorithm and the Gabriel graph are most efficient considering energy consumption. This is due to the fact that those algorithms try to chose relatively short edges, which usually means that the received signal strength is high. The selection of those strong links varies depending on the topology. It is based on an Euclidean metric for the EMST, on an energy-metric for the 1.1-energy spanner, and on proximity-based criteria for XTC and the Gabriel graph.

The number of packets that are sent is about 5 times higher for the EMST topology than for the $\text{RLS}_{\text{-86 dBm}}$ topology. However, since the transmission power grows cubically[5] with the distance, the energy that is conserved due to shorter distances to the neighbors outweighs the additional energy that is spend due to an increased number of sent packets.

The average transmission power is depicted in Figure 5.24(a), while a more detailed view on the overall energy consumption (using the DSDV algorithm) is depicted in Figure 5.24(b). We observe that the order of the topologies regarding the transmission power is identical to the one regarding the overall energy consumption. However, the transmission power on the left is given in dBm and thus on a logarithmic scale, in contrast to the total

[5] We use an attenuation coefficient of $\alpha = 3$ as described in Section 5.1.

energy consumption in joule. This indicates that the algorithms that are able to use a very low transmission power need to send more packets to finish the transmissions.



(a) Average transmission power

(b) Energy consumption using DSDV routing

Figure 5.24.: Average transmission powers in dBm are displayed on the left while a detailed view on the overall energy consumption for the transmissions using the DSDV routing algorithm can be seen on the right.

We can see that according to our measure the topologies that yield a good throughput performance, such as the RLS graph or the Yao graph, can not be considered energy-efficient as they do not allow a drastically reduced transmission power. Topology control algorithms that allow very low transmission power feature mostly short edges and hence they need more hops and achieve a lower throughput.

Therefore, depending on the application and the restrictions at hand, one must consider which topology fits best.

Note that our measure for overall energy consumption does not account for the time the nodes are powered on or idle. We assume that the node is idle after transmission (as it does not know when the last transmission finished). We also assumed that the energy needed for wireless communication device is solely based on the transmission power and that the transmission power can be adjusted continuously.

5.5. Topology Control and TDMA Schedules

In this section, we will consider joining TDMA scheduling and topology control. There are three main reasons to do this:

1. The length of a TDMA schedule can be considered an interference measure for topologies.

2. TDMA schedules yield sleep cycles that enable the nodes to conserve energy.

3. TDMA schedules are reported to yield a better performance than CSMA/CA.

We are now considering those reasons more detailed.

In the last section, the performance of traffic send between random sender-receiver pairs was the measure for the performance of the topology. As discussed in [WW07, page 109-112], the length of a transmission schedule can be a measure for interference in networks. In the literature, it is often assumed that each edge has to be scheduled once (or twice, to account for the bi-directionality). This may be sufficient for dense networks, however, this would prefer sparse network topologies immoderately, since the workload of the topologies may differ depending on the number of edges in the topology.

Hence, we chose to use traffic between random sender-receiver pairs to measure performance, as this yields a comparable amount of traffic that has to be transmitted through the network. Since the routing algorithms in the previous experiments were mostly balanced once sufficiently strong communication links are considered, a shortest path is chosen and every edge on this path is added to the set of edges that needs to be scheduled. With this, method we ensure that the load is comparable for all topologies.

As discussed in Section 2.2, the wireless network device may consume up to two thirds of the energy a wireless node requires. For CSMA/CA, the communication device must be enabled continuously to ensure availability. For TDMA schedules on the other hand, the time slots that are assigned to the node are known and hence the wireless network device can be disabled in time slots the node is not participating.

TDMA schedules are reported to yield a better performance on wireless mesh networks than CSMA/CA[BBS06a]. By joining the topologies considered earlier in this chapter with TDMA scheduling, we can compute a schedule for the workload on a typical wireless network topology.

5.5.1. Preliminaries, Parameters and Modifications

In this section, we describe how the TDMA schedules that are based on the topologies considered earlier in this chapter can be computed. Afterwards we describe the parameters that were adapted and the modifications in the ns-3 network simulator that were necessary for the experiments described in the next section.

To compute the schedules, we need to select communication links of the topology that are representative and use them as an input to the scheduling algorithms. This is done by using random sender-receiver pairs that need to transport a specified amount of data from the sender to the receiver. Each communication link that is on the[6] shortest path between a sender-receiver pair is added to the set of communication links that must be scheduled.

If a communication link is selected more than once, it is also added more than once, and its payload (i.e., the amount of data that must be transmitted over this communication link) is calculated accordingly. An example of the selected communication links as well as the amount of data that must be transmitted throughout the network is depicted in Figure 5.25. Communication links are drawn bolder the more sender-receiver pairs use this link.

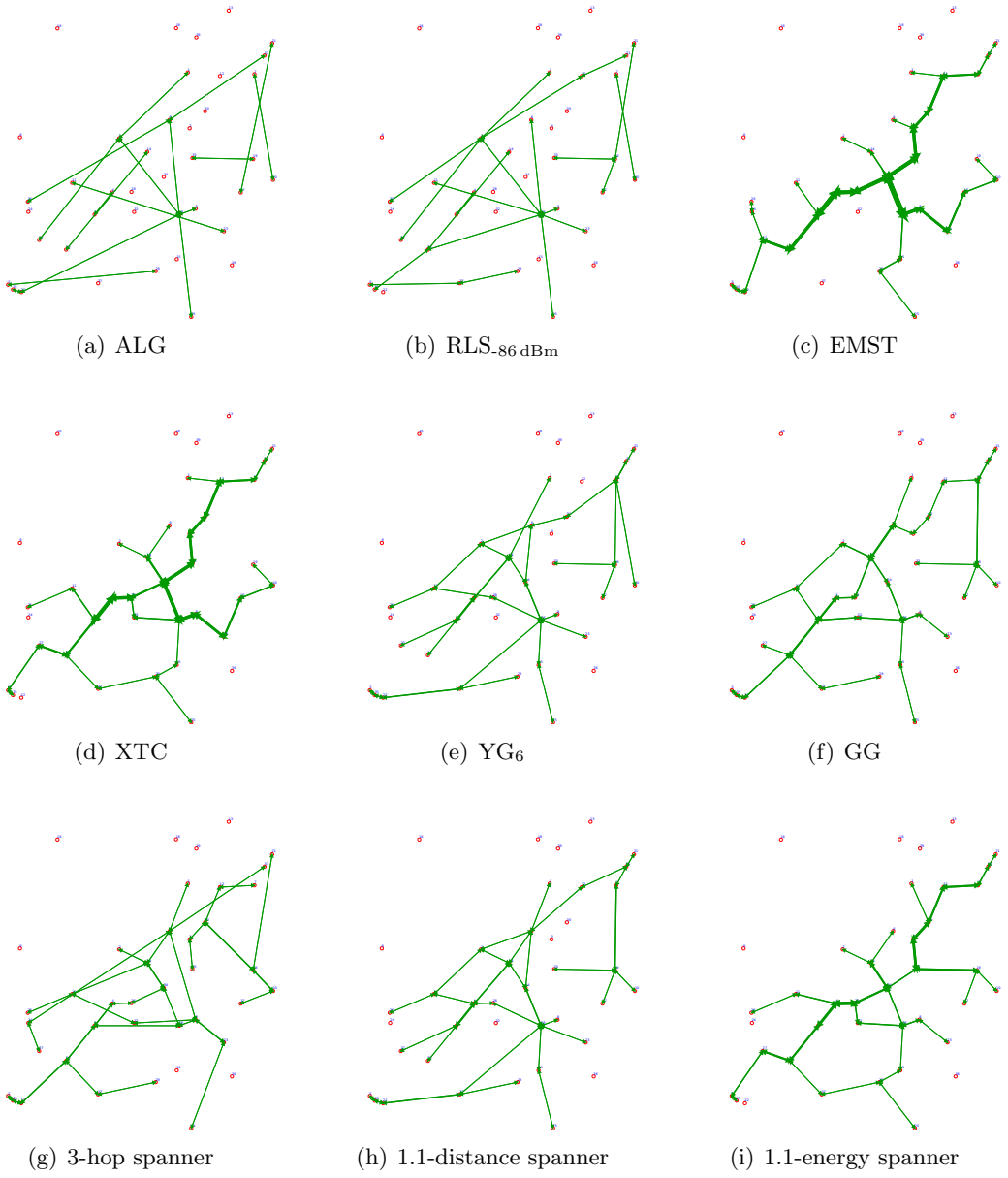
For these experiments, some modifications have been necessary in contrast to the experiments conducted in Section 5.4:

- ns-3 does not offer a MAC layer that implements TDMA schedules as mentioned in Chapter 4. As an implementation of the customization of the MAC layer was not considered an options, some adaptions had to be made. The issues regarding those adaptions are discussed in Section 4.2 and the modifications that have been made to ns-3 are described in Section 4.4.3. In short, the buffers have been minimized and the application that is used to send data according to the TDMA schedules is modified accordingly.

- In contrast to the previous experiments in this chapter, a constant data-rate manager is used in these experiments. The data-rate has been reduced from 24 Mbit/s (as in Chapter 4) to 6 Mbit/s. This is necessary since in many topologies long links are chosen and a data rate of 24 Mbit/s can not be achieved on those links.

- As in Chapter 4, we use the number of necessary time slots for comparison. Since we use a time slot length of 0.1 seconds, the number of time slots divided by ten equals the time needed to finish the transmissions.

[6] If between the sender and the receiver more than one shortest path exists, one may be chosen arbitrarily.



(a) ALG (b) RLS$_{-86\,\mathrm{dBm}}$ (c) EMST

(d) XTC (e) YG$_6$ (f) GG

(g) 3-hop spanner (h) 1.1-distance spanner (i) 1.1-energy spanner

Figure 5.25.: Shortest-path routes from the senders to the receivers. Each edge on each route is added to the set of transmissions that must be scheduled. Bolder links are used by more than one route and must be scheduled according to the number of sender-receiver pairs that use this edge.

- The data that must be transmitted over the links has been reduced to 1.5 MB. This has been changed to adapt the number of time slots needed to finish the transmission according to the reduced data-rate.
- Since all links that are chosen should be able to finish the transmission, links whose signal strength is too low to enable successful communication should not be considered. Hence, we restricted the set of available connection links (similar to the experiments in Section 5.4.2) to those links that have a signal strength of -90 dBm or above.
- In Chapter 4 each receiver needed only one `PacketSink` application to receive the packets and each sender needed only one `OnOffApplication` to send packets. In

this section, however, each node may need up to n instances of each application. Therefore, for both the `PacketSink` as well as the `OnOffApplication` identifiers are needed. Those identifiers are for both applications the address of the receiver consisting of its IP address along with the used port. Each node sends all the packet it sends to a unique port at the receiver and hence the combination of address and port of the receiver is sufficient to identify the communication link. A patch that modifies the application accordingly is given in Appendix A.1.

- Since the SINR thresholds are decreased as the data-rate is decreased, a new SINR threshold must be determined. Through experiments similar to those in Section 4.5.1, a threshold of 6 dBm has been determined.

Note that we do not require the time slots to be processed in such an order that if each slot is processed according to the schedule, a transmission would be possible from the sender to the receiver. This would restrict the scheduling algorithms to process the communication links in the order in which they are on the routes (i.e., the first link on the route must be scheduled before the second and so on). We allow the TDMA scheduling algorithms to schedule the communication links without considering this aspect.

Hence we can not directly compare the instances that use the CSMA/CA mechanism with those TDMA schedules regarding the delay of the data. However, after a number of iterations that is linear in the length of a hop-minimal shortest path from a sender to the corresponding receiver, the throughput that can be achieved using these TDMA schedules is comparable to the throughput achieved using the CSMA/CA mechanism.

5.6. Experiments II

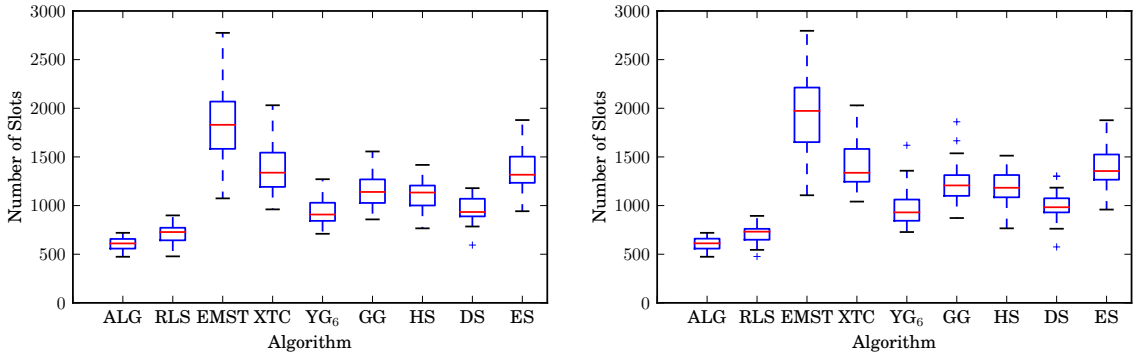
Since TDMA schedules allow sleep and duty cycles, they offer interesting possibilities to conserve energy in wireless sensor networks. However, so far, especially wireless communication that has high demands on throughput and transmission range does seldom utilize TDMA schedules. In the rest of this section, we will compute TDMA schedules based on the links chosen by the topology control algorithms. Using these schedules to simulate the transmission of data through the network, our experiments provide insights on how well the topologies perform for a different media access than CSMA/CA. This gives us a measure on the practicability of the topologies in TDMA-based networks.

We will first consider the number of time slots needed to finish the transmissions, which allows us to compare the topologies regarding the throughput that can be achieved using TDMA scheduling. We compare the time slots needed for both fixed transmission powers as well as variable transmission powers.

Again, for variable transmission powers the transmission power is determined individually for each node as the minimal transmission power such that the signal strength for any communication link selected by the topology control algorithm the signal strength is at least -90 dBm. Note that these communication links need not be selected as input for the scheduling algorithm.

For our experiments, we used 40 nodes that are randomly placed on a square area of 200×200 meters and 12 sender-receiver pairs that are used to compute the traffic that must be transmitted through the network. As described in the last section, for each sender-receiver pair those links that are on a shortest-path from the sender to the receiver are used as an input for the scheduling. Communication links that are on the shortest path between more than one sender-receiver pair are considered according to the number of shortest paths they are on.

In Figure 5.26 the number of slots the different topologies needed to finish the transmissions according to the TDMA schedule are depicted. The performance is shown for fixed transmission powers in Figure 5.26(a) and for variable transmission power in Figure 5.26(b). The GreedyBiSINR scheduling algorithm is used for both the fixed and the variable transmission powers.



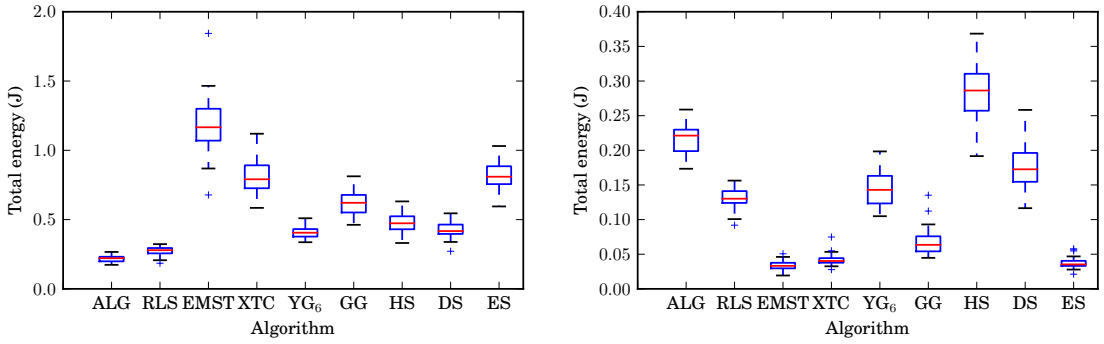
(a) Fixed transmission powers

(b) Variable transmission powers

Figure 5.26.: A comparison of the number of slots the topologies needed to finish all transmissions according to the computed TDMA schedules. The GreedyBiSINR scheduling algorithm is used and the topologie uses fixed or variable transmission powers. The transmission pairs are computed based on shortest paths from the random sender-receiver pairs in the topologies.

It can be observed that the results are basically identical. As the data rate is constantly set to 6 Mbit/s, the transmission does not suffer from a reduced SINR as it is the case for the variable data-rate, which is used in Section 5.4. Comparing the throughput of the topologies, we can see that the relative performance is similar to the relative performance in CSMA/CA. Only the ALG, which is actually the $\text{RLS}_{\text{-90 dBm}}$ algorithm since the set of input links is restricted to -90 dBm, achieves an even slightly better throughput than the $\text{RLS}_{\text{-86 dBm}}$. This is since -86 dBm is probably not the best threshold for the RLS algorithm when a constant data-rate of 6 Mbit/s is used. Since the data-rate is lower, weaker links can achieve a better performance in this setup than in the setup used in Section 5.4.2.

Since energy conservation is one major goal in topology control as well as in wireless sensor networks in general, we use the energy consumption assumptions described in Section 5.4.5 to calculate the energy consumption based on the transmission power and the number of packets that are sent. Based on the same experiment that were used in Figure 5.26, the overall energy consumption for the transmission according to the TDMA schedules is depicted in Figure 5.27.



(a) Fixed transmission powers

(b) Variable transmission powers

Figure 5.27.: Energy consumption of the transmissions in the network using TDMA schedules for fixed and variable transmission powers.

For fixed transmission powers, we can observe that similar to the transmission using the CSMA/CA mechanism in Section 5.4.5, the overall energy consumption is determined by the required number of time slots, i.e., the time needed to finish all transfers. Hence the ALG ($RLS_{-90\,dBm}$) and the RLS topologies dominate also for the overall energy consumption for fixed transmission powers. For variable transmission powers, those topologies that use mainly very short links are best if the energy consumption is considered. Namely those are the topologies based on the EMST, the 1.1-energy spanner and XTC. Similar to the transmissions using the CSMA/CA algorithm, this is due to the fact that the energy increases cubically with the distance.

We can conclude that the relative performance of the topologies for both the throughput and the overall energy consumption for instances using the TDMA scheduling are similar to the relative performance of the topologies using the CSMA/CA mechanism to manage medium access control.

However, as observed in Chapter 4, we can expect a higher throughput for TDMA schedules than for the CSMA/CA mechanism, in addition to the benefits regarding energy consumption.

5.7. Discussion

In this chapter, we first gave an overview on theoretical quality criteria that are often considered in connection with topology control. However, some of those criteria, such as the throughput or the interference, can not be easily described using mathematical models. For this reason, many topology control algorithms have only been analysed using mathematical models.

We then gave an introduction to some of these algorithms, as well as an analysis of the throughput performance and the energy consumption of the considered algorithms using the network simulator ns-3.

For some of the algorithms, it has not been clear for which parameters they perform best. For the hop, distance and energy spanners, the stretch factors that yields best throughput results have been determined. A simple but efficient approach that restricts the topology to the connection links that achieve a signal strength above a certain threshold is considered, and a good threshold value is determined as -86 dBm in Section 5.4.2. The more classical topology control algorithms have also been compared regarding their robustness towards weak links. We observed that, except for the ALG topology that allows all links, and the 2-hop spanner, all topologies aim at avoiding weak links since they restrict themselves to relatively good links whenever possible.

As for different applications a different number of sender-receiver pairs may be used, we considered how the topologies perform for an increasing amount of sender-receiver pairs and hence an increasing amount of data that must be transmitted over the network. Our experiments show that for rather dense networks, most topologies can cope with an increased amount of data. The throughput for the topologies is mainly restricted by the average number of hops that must be used in the topologies. For few hops a high throughput can be achieved, while for many hops the throughput is considerably lower. A simple restriction of the signal strength in the RLS graph, the YG_6 and the 1.1-distance spanner yield the best results for all considered numbers of pairs.

For a denser setup, however, the situation is not so clear. Depending on the routing algorithm, some topologies achieve only relatively unstable results. Generally, it can be said that the sparse topologies such as those based on the XTC, the 1.1-energy spanner and the EMST, achieve the most stable results. This is due to the relatively short links they use.

As for only two different densities not all properties of the topologies could be considered, we studied the performance of the topologies for various different-sized deployment areas. We found that dense topologies such as the RLS graph, the YG_6 or the 1.1-distance

spanner achieve the best throughput-performance for dense networks. However, for sparse networks the rather dense topologies are often outperformed by sparse topologies such as the topologies based on XTC or the 1.1-energy spanner. This implies that the throughput performance is highly dependent on the number of hops in general. However, if the signal strength is relatively weak for many communication links, the improvement achieved by a restriction on only those communication links with sufficient signal strength outperforms the drawbacks of communication over many hops. Regarding the transmission power, we find that if variable transmission powers are not enabled, the topologies that achieve the highest throughput are most energy-efficient. This is mostly due to the fact that those topologies use less hops and hence conserve transmission energy.

If variable transmission powers are considered, however, topologies that use mainly short links achieve a good overall energy consumption. This is since those topologies allow the nodes of the network to reduce the transmission powers dramatically. Since the energy grows cubically with the distance, this usually results in lower consumed energy even if more hops are needed. For variable transmission powers, the EMST, followed by the 1.1-energy spanner and XTC achieve the lowest overall energy consumption.

Combining topology control with TDMA scheduling shows that the relative results achieved with CSMA/CA are similar to those that can be achieved when scheduling is used for medium access. Our results show that the topologies based on a restriction of the link strengh, followed by the YG_6 and the 1.1-distance spanner, allow the highest throughput while the topologies based on EMST, 1.1-energy spanner and XTC are most energy-efficient.

Regarding the question of whether sparse topologies reduce interference or not, it is hard to make definitive statements since there is no commonly accepted measure for interference. We have observed in the experiments in this chapter that dense topologies usually achieve a higher throughput for dense networks, while sparse topologies are more robust towards sparse networks. This is mostly due to the number of hops that are used and not directly related to interference. For sparse topologies, the number of hops required to connect two nodes is relatively high, and hence the throughput that may be achieved is considerably reduced. The performance of multi-hop communication is, at least partially, reduced due to interference between the nodes. Considering the transmission power however, we found that the transmission power can be reduced significantly for sparse topologies and hence the interference on other nodes is reduced. However, due to the increased number of hops this reduction of interference does not yield a better throughput.

6. Conclusion

In wireless sensor networks, where energy is a valuable and scarce resource, it is important to communicate efficiently, since inefficient communication consumes additional energy. In this work, we studied two techniques that can help to improve the efficiency of communication in wireless sensor networks, topology control and TDMA scheduling. Topology control restricts communication to a subset of the possible communication links that are assumed to achieve more efficient communication. Scheduling can be used to manage medium access by dividing the time in time slots and assigning transmissions to time slots such that they can transmit data simultaneously. Schedule-based medium access can reduce interference and thus improve the throughput.

So far, many topology control algorithms have mainly been analyzed theoretically. One of the goals of this thesis was, to study the performance of those algorithms based on simulations in a network simulator. Also, based on these simulations, the influence of certain properties of the topologies on the throughput and the energy efficiency is studied. We considered performance in terms of time needed to transmit the traffic generated by random sender-receiver pairs as well as in terms of the energy consumed during the transmission of this data. Both performance measures are studied for communication with fixed as well as with variable transmission powers.

We observed that for rather simple and static routing algorithms, topology control can help to considerably improve performance. The build-in routing algorithms such as OLSR and DSDV, in contrast, automatically avoid relatively weak links that appear to be broken by themselves. This is, by itself, a form of topology control as the routing algorithm selects a subset of the available communication links[1]. However, for all routing algorithms a simple restriction to communication links that exceed a certain signal strength, which is done in the RLS topology, improves the throughput. More refined topology control algorithms often select communication links in a way that significantly increases the average number of hops on the route between two nodes. We observed that the number of hops is a very important property of the topologies if we consider the throughput performance, as the time needed to finish transmissions increases as the average number of hops does. This is true for communication using CSMA/CA as well as for TDMA schedules to organize medium access.

As the density of the networks decreases, we observed that a general restriction on links up to a certain threshold does not yield the desired results as networks are mostly not connected anymore. For such networks, many rather dense topologies can not produce

[1] Note that all communication links considered have at least once been able to successfully transmit a packet.

stable results anymore as relatively weak links may be selected. For sparse networks, we found that topologies that select only few but relatively strong links achieve the most stable results. Namely those are the topologies based on EMST, the XTC or the 1.1-energy spanner.

If we consider the energy consumption for fixed transmission powers, those topologies that achieve the highest throughput, which coincides with the lowest average number of hops, are most energy-efficient according to our measure[2]. Again, this is the RLS graph, followed by the Yao graph and a 1.1-distance spanner. For variable transmission powers, however, the topologies that consist mainly of very short links and hence allow a low transmission power achieve the best energy efficiency. The drawback of more transmitted packets is more than outweighed by the reduction of transmission power.

Another motivation of this thesis in addition to a comparison of the considered topology control algorithms was a further study of whether interference is minimized by sparse topologies, as it has often been assumed. Based on the results of Chapter 5, we can say that since the transmission power can be reduced significantly for the considered sparse topologies, the interference on other nodes is reduced. However, we have also observed that the throughput performance of the networks highly depends on the average number of hops between two nodes in the network. For sparse topologies, the average number of hops is relatively high and the throughput performance low. Hence we can conclude that even though the interference is reduced, this does not yield a better throughput performance. However, if we focus on energy efficiency, we observed that sparse topologies, which mostly rely on few but relatively strong links, can minimize the energy consumption significantly.

TDMA schedules offer some benefits regarding energy consumption by enabling the use of sleep and duty cycles to the communication links and hence enabling the nodes to conserve energy in time slots, in which they are not participating. The actual performance of TDMA schedules has been compared to the IEEE 802.11a CSMA/CA mechanism. We found that even though the MAC layer implemented in ns-3 is not optimized towards TDMA schedules, the TDMA schedules yield a considerably better performance regarding the throughput. We did also compare two slightly different interference models. The general SINR model, which accounts only for interference caused by senders and ensures the SINR constraint only for receivers, and the bi-directional SINR model, which accounts for interference caused by all nodes and ensures the SINR constraint for senders as well as receivers. In our experiments, we found that schedules based on the bi-directional SINR model yield slightly better results for a narrow range of SINR threshold values.

In the literature, applying TDMA schedules to topologies has mostly been considered either theoretically [Mos06] or as a metric for the topologies [WW07, page 108-112]. In our simulations, we applied TDMA schedules to transmission pairs that simulated the traffic of sender-receiver pairs. We observed that the relative performance of the topologies using the TDMA schedules is similar to the relative performance using the CSMA/CA mechanism. Thus, a restriction on links up to a certain threshold enables the highest throughput while the rather sparse topologies that restrict communication to very short communication links, such as those based on the EMST, the XTC or the 1.1-energy spanner, resulted in the lowest overall energy consumption.

6.1. Outlook

We observed in this thesis that topology control is vital for very simple, static routing algorithms. Routing algorithms that have been designed for mobile wireless ad hoc networks are already relatively robust against weak communication links. However, several topology control algorithms improve the throughput performance even for those routing algorithms. The topology that simply discards all weak links achieves the best throughput performance. Unfortunately, this topology does not necessarily yield a connected network.

[2]Which is based on the number of sent packets and the transmission power.

Finding a threshold such that connectivity persists while the throughput is maximized is interesting, however, finding such a threshold locally would be desirable. Also, integrating a restriction on strong links in routing algorithms is interesting since we observed an improvement for such a combination in our experiments.

Another interesting field is interference minimization. We observed in this thesis that interference can be reduced by sparse topologies, however, while significantly decreasing the throughput. In the past years, new measures of interference have been introduced (most popular are the sender-centric and the receiver-centric interference models, cf. [vRWZ09]) and topologies proposed that minimize interference according to those measures [MNL05, LZLD08, YDE11, LTWL11]. In [YDE11], the proposed heuristic that reduces receiver-centric interference is compared to a minimum spanning tree regarding the energy consumption in a simulation-based analysis. However, a further assessment of the implication of these new measures of interference by the means of throughput maximization as well as a more thorough analysis regarding the energy consumption is another interesting research direction.

7. Deutsche Zusammenfassung

Drahtloskommunikation ist eine der bedeutendsten Neuerungen unserer Zeit. Geräte, die drahtlose Kommunikation ermöglichen, sind inzwischen kostengünstig und können vielseitig eingesetzt werden. Sensoren, also Geräte mit denen Eigenschaften der Umwelt erfasst und digital verarbeitet werden, können sich mittels Drahtloskommunikation mit anderen Sensoren oder einer Basisstation zu einem sogenannten drahtlosen Sensornetzwerk zusammenschließen. Sensoren mit einer entsprechenden Kommunikationseinheit werden Sensorknoten genannt.

Mittels drahtloser Sensornetzwerke können viele alltägliche Aufgaben effizient gelöst werden. So können zum Beispiel in einem Krankenhaus Patienten mit Sensoren ausgestattet werden, die aufgezeichnete Informationen drahtlos an eine zentrale Basisstation oder direkt an Ärzte und Personal weiterleiten. Ein weiteres mögliches Einsatzszenario sind Krisengebiete, in denen die Infrastruktur weitestgehend zerstört ist. Hier können Sensorknoten verwendet werden um automatisch ein drahtloses Netzwerk aufzubauen und so gezielt Informationen auszutauschen.

Für die Realisierung dieser drahtlosen Netzwerke gibt es viele von der jeweiligen Anwendung abhängige Einschränkungen, allerdings auch einige, die generell für drahtlose Sensornetzwerke gelten. Zum einen sind die Knoten nicht mit der Infrastruktur verbunden und müssen daher eine möglichst lange Laufzeit ermöglichen und zum anderen sollen sie klein und kostengünstig sein. Da kleine und kostengünstige Sensorknoten meist keine entsprechend leistungsstarke Batterie aufweisen, sollte möglichst wenig Energie verbraucht werden. Da drahtlose Kommunikation den Hauptteil der Energie verbraucht, muss darauf geachtet werden, dass die Kommunikation möglichst effizient abläuft.

Die Kommunikation in drahtlosen Sensornetzwerken war in den letzten Jahren ein aktives Forschungsgebiet. Eines der interessantesten Gebiete in diesem Forschungsfeld umfasst die Topologiekontrolle, bei der aus der Menge der möglichen Kommunikationspartner eine Teilmenge so ausgewählt werden soll, dass Interferenz zwischen den Knoten minimiert und energieeffiziente Kommunikation ermöglicht wird. Viele der aktuell vorgeschlagenen Topologiekontroll-Algorithmen wurden in erster Linie theoretisch analysiert. Dabei wird häufig angenommen, dass dünne Topologien (also Topologien mit wenig Kommunikationspartnern pro Knoten) Interferenz minimieren. Ein Ziel dieser Diplomarbeit ist es, diesen Zusammenhang weiter zu untersuchen. Hierzu werden wir die Leistung dieser Algorithmen sowohl in Bezug auf den erzielbaren Datendurchsatz als auch in Bezug auf die Energieeffizienz mit dem Netzwerksimulator ns-3 untersuchen.

Ein weiteres interessantes Forschungsgebiet ist es, den Zugriff auf das Kommunikationsmedium derart zu koordinieren, dass Interferenz und Kollisionen vermieden werden. In

dieser Diplomarbeit wurden bestehende Algorithmen zur Berechnung von Ablaufplänen für den Medienzugriff im Zeitmultiplexverfahren, sogenannte TDMA Schedules, mittels Simulationen im Netzwerksimulator mit dem CSMA/CA Verfahren verglichen. Das CSMA/CA Verfahren wird im IEEE 802.11a Standard für Medienzugriff und Kollisionsvermeidung eingesetzt. Abschließend wurde die Kombination von Topologiekontrolle und TDMA Zugriffsverfahren untersucht.

Um den Datendurchsatz in Topologiekontroll-Algorithmen zu bestimmen, wird die Zeit gemessen, die benötigt wird um Daten von einer Menge von Sender-Empfänger-Paaren durch das Netz zu senden. Die Sender-Empfänger-Paare sind hierbei zufällig gewählt und der durch das Netzwerk gewählte Weg wird durch Routenplanungs-Algorithmen festgelegt.

Wir konnten in unseren Experimenten feststellen, dass für eher dichte Netzwerke, also Netzwerke mit vielen Sensorknoten auf einer recht kleinen Fläche, eine Einschränkung der möglichen Kommunikationsverbindungen auf Verbindungen mit hoher Signalstärke den höchsten Durchsatz ermöglicht. Für eher dünne Netzwerke ist dieses Verfahren zum Teil nicht mehr geeignet, da die berechnete Topologie nicht mehr unbedingt zusammenhängend ist. Hier liefern eher dünne Topologien, wie die denen ein Energie-Spanner oder der XTC Algorithmus zugrunde liegen die besten Ergebnisse.

Um den Energieverbrauch für die Übertragung der Daten zwischen Sender-Empfänger-Paaren des drahtlosen Sensornetzwerkes zu schätzen wurde angenommen, dass die Anzahl der gesendeten Datenpakete maßgeblich für den gesamten Energieverbrauch ist. Unter dieser vereinfachenden Annahme zeigen unsere Experimente, dass sich der gesamte Energieverbrauch für feste Sendeleistungen analog zur benötigten Zeit zum Beenden der Übertragung verhält. Für variable Sendeleistungen zeigt sich ein anderes Bild. Hier dominieren die Topologien, die ausschließlich recht kurze Kanten wählen und daher die Sendeleistung massiv senken können. Dies ist vor allem die Topologie zum euklidischen minimalen Spannbaum, aber auch die Topologien, denen ein Energie-Spanner, der XTC Algorithmus oder der Gabriel Graph zugrunde liegt.

Die Topologien, die einen hohen Datendurchsatz ermöglichen, können leider nicht als energieeffizient bezeichnet werden.

Für die Untersuchung der Algorithmen zur Berechnung von TDMA Schedules, sogenannte Scheduling-Algorithmen, mussten zunächst einige Anpassungen am verwendeten Netzwerksimulator ns-3 vorgenommen werden, da der Simulator Medienzugriff nach Zeitmultiplexverfahren nicht implementiert. Durch einige Änderungen an den Simulatorparametern und dem Zeitmultiplexverfahren kann ein solches Verfahren allerdings über den im Simulator implementierten Medienzugriff simuliert werden.

Für den Vergleich der Scheduling-Algorithmen mit dem CSMA/CA Verfahren wurde eine Menge an Übertragungen zufällig auf einem Gebiet verteilt. Nun wurde unter Berücksichtigung eines Modells zur Einhaltung von Signal-zu-Rausch-Verhältnissen jede Übertragung einem Zeitslot zugeordnet. Durch die Berücksichtigung physikalischer Gegebenheiten nach dem Signal-zu-Rausch-Verhältnis wird sichergestellt, dass die Übertragungen innerhalb eines Zeitslots alle gleichzeitig bearbeitet werden können. Mittels des Netzwerksimulators wird nun ein Zeitslot nach dem anderen abgearbeitet bis die Übertragungen alle beendet sind. Unsere Ergebnisse zeigen, dass die Übertragungen schneller beendet sind wenn TDMA Schedules benutzt werden; TDMA Schedules also einen höheren Datendurchsatz erlauben als das CSMA/CA Verfahren.

Abschließend wurde die Kombination von Topologiekontroll- und Scheduling-Algorithmen untersucht. Hierfür wurden zunächst Topologien für das Netzwerk berechnet und anschließend die Knotenverbindungen, die zur Übertragung der Daten zwischen zufälligen Sender-Empfänger-Paaren benutzt werden, mittels eines Scheduling-Algorithmus in Zeitslots eingeteilt. Um den Datendurchsatz zu bestimmen wurde erneut die Zeit gemessen, die vom Netzwerksimulator benötigt wurde um alle Übertragungen gemäß der Einteilung in Zeitslots abzuarbeiten. Hier konnte ähnlich wie zuvor festgestellt werden, dass die

Topologie, welche nur eine Einschränkung der möglichen Kommunikationsverbindungen auf Verbindungen mit hoher Signalstärke vornimmt, gute Ergebnisse erzielt. Ähnlich wie für das CSMA/CA Verfahren orientiert sich der erwartete Energieverbrauch für feste Sendeleistungen auch für TDMA Schedules stark an der benötigten Zeit um die Übertragungen abzuschließen. Daher konnte der niedrigste erwartete Energieverbrauch für feste Sendeleistungen durch die Topologie, welche die Kommunikationsverbindungen auf signalstarke Verbindungen einschränkt, erreicht werden. Für variable Sendeleistungen bieten die Topologien, welche eine starke Reduzierung der Sendeleistung ermöglichen, erneut die besten Ergebnisse, allen voran die auf dem euklidischen minimalen Spannbaum basierende Topologie.
Eines der Ziele dieser Diplomarbeit war es, den Zusammenhang zwischen dünnen Topologien und Interferenz in drahtlosen Sensornetzwerken weiter zu untersuchen. Basierend auf den Ergebnissen in Kapitel 5 können wir sagen, dass der über eine Topologie erzielbare Durchsatz nicht direkt von der Interferenz abhängt, sondern mit einigen Eigenschaften wie der Anzahl der für eine Übertragung notwendigen Knoten eng verbunden ist. Durch eine Verringerung der Sendeleistungen reduzieren die betrachteten dünnen Topologien die Interferenz auf andere Knoten, es ergibt sich aber durch die Erhöhung der durchschnittlich nötigen Knoten für die Verbindung zweier Knoten auch ein wesentlich verringerter Datendurchsatz. Allerdings zeigen unsere Experimente klar, dass sich dünne Topologien in Bezug auf den Energieverbrauch lohnen, da in dünnen Topologien hauptsächlich kurze und signalstarke Kommunikationsverbindungen ausgewählt werden.

A. Appendix

A.1. Patches to ns-3

```
diff -r org/src/internet/model/tcp-socket-base.cc mod/src/
    internet/model/tcp-socket-base.cc
38d39
> #include "ns3/random-variable.h"
1240c1238,1239
<           m_rto = m_cnTimeout * backoffCount;
---
>           UniformVariable a;
>           m_rto = m_cnTimeout * backoffCount * a.GetValue
    (0.99,1.01);
```

Figure A.1.: Adding random effects to the back-off timeout of the TCP implementation, as described in Section 5.3.1.

```
diff -r org/src/applications/model/onoff-application.cc mod/src/
    applications/model/onoff-application.cc
120a121,126
> Address
> OnOffApplication::GetPeer (void) const
> {
>   NS_LOG_FUNCTION (this);
>   return m_peer;
> }
238a245,250
> void OnOffApplication::StartFor(Time startingTime, Time
    duration)
> {
>   CancelEvents();
>   m_startStopEvent = Simulator::Schedule (startingTime, &
    OnOffApplication::StartSending, this);
>   m_startStopEvent = Simulator::Schedule (startingTime+duration
    , &OnOffApplication::StopSending, this);
> }
242a255,258
>   if (!m_sendEvent.IsExpired ())
>   {
>     CancelEvents();
>   }
diff -r org/src/applications/model/onoff-application.h mod/src/
    applications/model/onoff-application.h
109a110,113
>   //public interface for schedule control
>   void StartFor(Time startingTime, Time duration);
>
>   Address GetPeer (void) const;
diff -r src//applications/model/onoff-application.h src/
    applications/model/onoff-application.h
109a110,113
>   //public interface for schedule control
>   void StartFor(Time startingTime, Time duration);
>
>   Address GetPeer (void) const;
```

Figure A.2.: Modifications to the ns-3 `OnOffApplication` as described in Section 4.4.3.

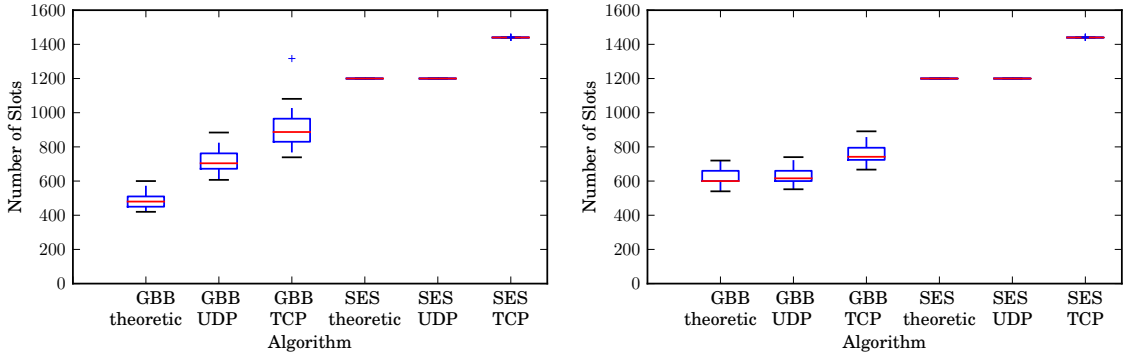
```
diff -r src//applications/model/packet-sink.cc src/applications/
    model/packet-sink.cc
197a198,204
> Address
> PacketSink::GetLocalAddress(void) const
> {
>   NS_LOG_FUNCTION (this);
>   return m_local;
> }
>
diff -r ns-3.13/src//applications/model/packet-sink.h src/
    applications/model/packet-sink.h
89a90
>   Address GetLocalAddress (void) const;
```

Figure A.3.: Modifications to the ns-3 `PacketSink` to allow an identification of the application as described in Section 5.5.1.

```
diff -r org/src/aodv/model/aodv-routing-protocol.cc mod/src/aodv/
    model/aodv-routing-protocol.cc
239a240,247
> std::map<Ipv4Address, std::map<Ipv4Address, double> >
    RoutingProtocol::topologyMap;
>
> void
> RoutingProtocol::setTopologyMap(std::map<Ipv4Address, std::map<
    Ipv4Address, double> > tm)
> {
>   RoutingProtocol::topologyMap = tm;
> }
>
913a922,928
>
>   if (RoutingProtocol::topologyMap[sender][receiver] != 1)
>   {
>     return;
>   }
>   else
>     NS_LOG_DEBUG("AODV node " << this << " received a AODV
    packet from " << sender << " to " << receiver);
diff -r ./ns-allinone-3.13-clean/ns-3.13/src//aodv/model/aodv-
    routing-protocol.h topo-ns-3.13/ns-3.13/src/aodv/model/aodv-
    routing-protocol.h
91a92,93
>   static void setTopologyMap(std::map<Ipv4Address, std::map<
    Ipv4Address, double> > tm);
>
92a95,96
>
>   static std::map<Ipv4Address, std::map<Ipv4Address, double> >
    topologyMap;
```

Figure A.4.: Modifications to the build-in routing algorithm DSDV. Similar changes apply for the OLSR routing algorithm. These modifications are described in Section 5.3.1.

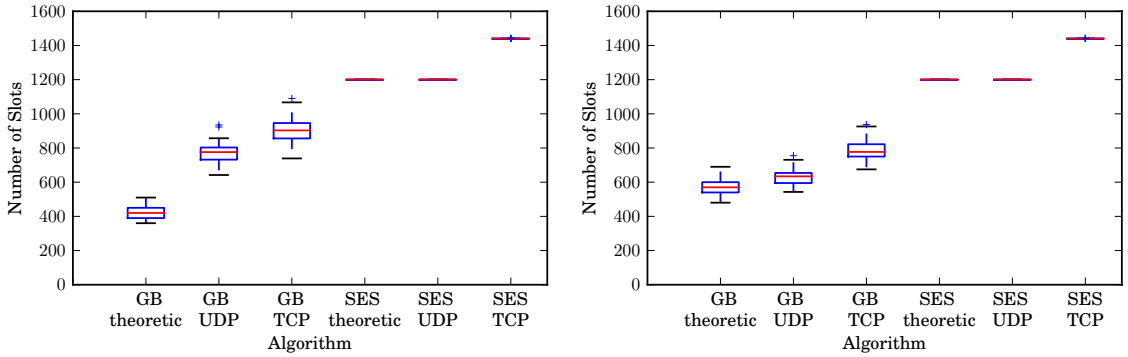
A.2. Additional Figures: TDMA vs. Separate Scheduling



(a) SINR threshold of 8 dB

(b) SINR threshold of 18 dB

Figure A.5.: Comparing the TDMA schedules computed by the GreedyBiBuffer algorithm to a TDMA schedule that assigns each transmission its own slot. The SINR threshold for the scheduling algorithms is set to 8 dB on the left and 18 dB on the right. The setup is according to Section 4.5.2

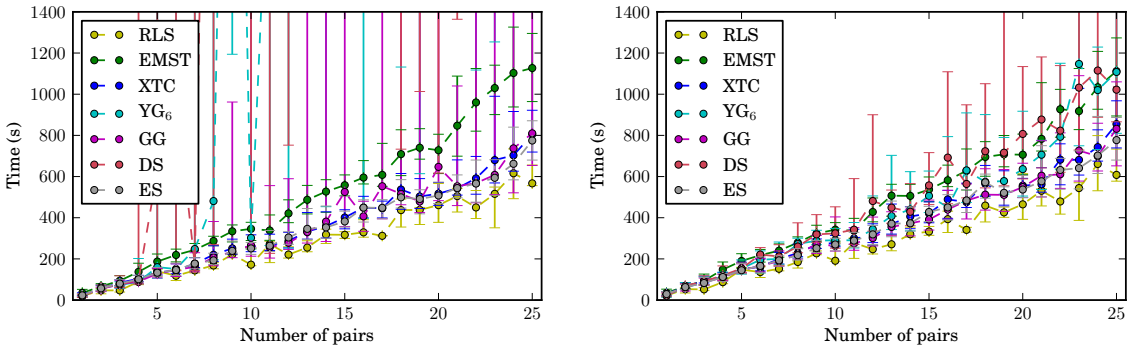


(a) SINR threshold of 8 dB

(b) SINR threshold of 18 dB

Figure A.6.: Comparing the TDMA schedules computed by the GreedyBuffer algorithm to a TDMA schedule that assigns each transmission its own slot. The SINR threshold for the scheduling algorithms is set to 8 dB on the left and 18 dB on the right. The setup is according to Section 4.5.2

A.3. Additional Figures: Increasing the workload



(a) Shortest-path routing

(b) DSDV routing

Figure A.7.: The time needed to finish the transmissions for the relatively sparse topologies on an area of 400×400 meters.
We can see that primarily for the static shortest-path routing the larger deployment area causes that some transmissions can not or only slowly finish for many instances. For the DSDV algorithm the transmissions can be finished for almost all instances. However, especially the 1.1-distance spanner shows a high fluctuation.

List of Acronyms

6LoWPAN	IPv6 over low power WPAN
ACK	Acknowledgement flag; used for short replies to acknowledge the reception of a packet
ALG	All Links Graph
AODV	Ad-hoc On-demand Distance Vector
API	Application Programming Interface
ARP	Address Resolution Protocol
CCA	Clear Channel Assessment
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSMA	Carrier Sense Multiple Access
CTS	Clear to Send
DCF	Distributed Coordination Function
DSDV	Destination-Sequenced Distance Vector
EMST	Euclidean Minimal Spanning Tree
GG	Gabriel Graph
GNU	GNU's Not Unix
GTNetS	Georgia Tech Network Simulator
IEEE	Institute of Electrical and Electronics Engineers
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
IP	Internet Protocol
LTE	Long Term Evolution
MAC	Medium Access Control
OLSR	Optimized Link State Routing
OSI	Open Systems Interconnection
QoS	Quality of Service

RFC	Request for Comment; memorandum on internet standards and protocols
RLS	Restricted Link Strength Graph
RNG	Relative Neighborhood Graph
RTS	Request to Send
$SINR_G$	Geometric Signal to Interference and Noise Ratio
SINR	Signal to Interference and Noise Ratio
STL	Standard Template Library
SYN	Synchronize flag; used for the first packets of a connection to synchronize sequence numbers
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
UAN	Underwater Acoustic Network
UDP	User Datagram Protocol
WAHN	Wireless Ad Hoc Network
WSN	Wireless Sensor Network
WiFi	Wireless Fidelity (usually a IEEE 802.11 wireless network)
WiMAX	Worldwide Interoperability for Microwave Access
YANS	Yet Another Network Simulator
YG	Yao Graph
dBm	Decibel of power relative to 1mW
dB	Decibel
mW	Milliwatt
ns-2	Network Simulator ns-2

List of Figures

List of Tables

List of Algorithms

Bibliography

[ASSC02] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, *Wireless sensor networks: a survey*, Computer Networks **38** (2002), 393–422.

[BBS06a] Gurashish Brar, Douglas M. Blough, and Paolo Santi, *Computationally efficient scheduling with the physical interference model for throughput improvement in wireless mesh networks*, Proceedings of the 12th annual international conference on Mobile computing and networking (New York, NY, USA), MobiCom '06, ACM, 2006, pp. 2–13.

[BBS06b] Gurashish Singh Brar, Douglas M. Blough, and Paolo Santi, *Computationally efficient scheduling with the physical interference model for throughput improvement in wireless mesh networks*, Proceedings of the 12th annual international conference on Mobile computing and networking (New York, NY, USA), MobiCom '06, ACM, 2006, pp. 2–13.

[BLRS03] Douglas M. Blough, Mauro Leoncini, Giovanni Resta, and Paolo Santi, *The k-neigh protocol for symmetric topology control in ad hoc networks*, Proceedings of the 4th ACM international symposium on Mobile ad hoc networking and computing (New York, NY, USA), MobiHoc '03, ACM, 2003, pp. 141–152.

[Buc08] Kevin Buchin, *Minimizing the maximum interference is hard*, arXiv/0802.2134 (2008).

[BvRWZ04] Martin Burkhart, Pascal von Rickenbach, Roger Wattenhofer, and Aaron Zollinger, *Does topology control reduce interference?*, Proceedings of the 5nd ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc '04, ACM, 2004, pp. 9–19.

[CJ03] T. Clausen and P. Jacquet, *Optimized link state routing protocol (OLSR)*, RFC 3626 (Proposed Standard), 2003.

[CLRS09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, *Introduction to algorithms*, 3 ed., The MIT Press, 2009.

[DANB00] A. Doufexi, S. Armour, A. Nix, and D. Bull, *A comparison of HIPERLAN/2 and IEEE 802.11a physical and MAC layers*, Symposium on Communications and Vehicular Technology, SCVT '00, 2000, pp. 14–20.

[EE04] Tamer A. ElBatt and Anthony Ephremides, *Joint scheduling and power control for wireless ad hoc networks*, IEEE Transactions on Wireless Communications **3** (2004), no. 1, 74–85.

[Erg04] Sinem C. Ergen, *ZigBee/IEEE 802.15.4 Summary*, Tech. report, EECS Berkeley, September 2004.

[FR09] Yasir Faheem and Jean Louis Rougier, *Loop avoidance for fish-eye OLSR in sparse wireless mesh networks*, Proceedings of the 6th international conference on Wireless On-Demand Network Systems and Services (Piscataway, NJ, USA), WONS '09, IEEE Press, 2009, pp. 215–218.

[Gar07] Vijay Garg, *Wireless communications and networking*, 1st ed., Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007.

[GGH+01] Jie Gao, Leonidas J. Guibas, John Hershberger, Li Zhang, and An Zhu, *Geometric spanner for routing in mobile networks*, Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing (New York, NY, USA), MobiHoc '01, ACM, 2001, pp. 45–55.

[GH01] Jimmi Grönkvist and Anders Hansson, *Comparison between graph-based and interference-based STDMA scheduling*, Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking and computing (New York, NY, USA), MobiHoc '01, ACM, 2001, pp. 255–258.

[GOW07] Olga Goussevskaia, Yvonne Anne Oswald, and Rogert Wattenhofer, *Complexity in geometric SINR*, Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing, MobiHoc '07, ACM, 2007, pp. 100–109.

[gtn08] *Georgia tech network simulator GTNetS*, Online: http://www.ece.gatech.edu/research/labs/MANIACS/GTNetS/, 2008.

[GWHW09] Olga Goussevskaia, Roger Wattenhofer, Magnús M. Halldórsson, and Emo Welzl, *Capacity of arbitrary wireless networks*, INFOCOM, 2009, pp. 1872–1880.

[HLR08] T. R. Henderson, M. Lacage, and G. F. Riley, *Network simulations with the ns-3 simulator*, ACM SIGCOMM'08, ACM, August 2008, p. 527.

[HM11] Magnús M. Halldórsson and Pradipta Mitra, *Nearly optimal bounds for distributed wireless scheduling in the SINR model*, ICALP (2), 2011, pp. 625–636.

[IEE07] IEEE Std 802.11-2007, *IEEE standard for information technology-telecommunications and information exchange between systems-local and metropolitan area networks-specific requirements - part 11: Wireless lan medium access control (MAC) and physical layer (PHY) specifications*, IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999) (2007), C1–1184.

[Kes11] Thomas Kesselheim, *A constant-factor approximation for wireless capacity maximization with power control in the SINR model*, Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '11, SIAM, 2011, pp. 1549–1559.

[KPX07] Iyad A. Kanj, Ljubomir Perkovic, and Ge Xia, *An optimal localized approximation scheme for euclidean MST*, 17th Fall Workshop on Computational and Combinatorial Geometry, FWCG '07, 2007.

[KV10] Thomas Kesselheim and Berthold Vöcking, *Distributed contention resolution in wireless networks*, Proceedings of the 24th international conference on Distributed computing (Berlin, Heidelberg), DISC '10, Springer-Verlag, 2010, pp. 163–178.

[LH06] Mathieu Lacage and Thomas R. Henderson, *Yet another network simulator*, Proceeding from the 2006 workshop on ns-2: the IP network simulator (New York, NY, USA), WNS2 '06, ACM, 2006.

[LHS05] Ning Li, Jennifer C. Hou, and Lui Sha, *Design and analysis of an mst-based topology control algorithm*, IEEE Transactions on Wireless Communications **4** (2005), no. 3, 1195–1206.

[LTWL11] Tiancheng Lou, Haisheng Tan, Yuexuan Wang, and Francis C. M. Lau, *Minimizing average interference through topology control*, Proceedings of the 7th International Symposium on Algorithms for Sensor Systems, Wireless Ad Hoc Networks and Autonomous Mobile Entities, ALGOSENSORS '11, 2011, pp. 115–129.

[LZLD08] Yongzhen Liu, Xinming Zhang, Qiong Liu, and Shifang Dai, *A hybrid interference model-based topology control algorithm*, Proceedings of the 4th International Conference on Networked Computing and Advanced Information Management - Volume 01 (Washington, DC, USA), NCM '08, IEEE Computer Society, 2008, pp. 42–46.

[MKHC07] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, *Transmission of IPv6 packets over IEEE 802.15.4 networks*, RFC 4944 (Proposed Standard), September 2007.

[MNL05] Kousha Moaveni-Nejad and Xiang-Yang Li, *Low-interference topology control for wireless ad hoc networks*, ACM Wireless Networks, IEEE Press, 2005.

[Mos06] Thomas Moscibroda, *Topology control meets SINR: The scheduling complexity of arbitrary topologies*, Proceedings of the 7nd ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc '06, ACM, 2006, pp. 310–321.

[MWW06] Thomas Moscibroda, Roger Wattenhofer, and Yves Weber, *Protocol design beyond graph-based models*, 5th Workshop on Hot Topics in Networks, HotNets '06, November 2006.

[NCc+11] Hemanth Narra, Yufei Cheng, Egemen K. Çetinkaya, Justin P. Rohrer, and James P. G. Sterbenz, *Destination-sequenced distance vector (DSDV) routing protocol implementation in ns-3*, Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques, SIMUTools '11, ICST, 2011, pp. 439–446.

[NKSK02] Swetha Narayanaswamy, Vikas Kawadia, R. S. Sreenivas, and P. R. Kumar, *Power control in ad-hoc networks: Theory, architecture, algorithm and implementation of the COMPOW protocol*, in European Wireless Conference, 2002, pp. 156–162.

[ns211] *Network simulator ns-2*, Online: http://nsnam.isi.edu/nsnam/, Nov 2011.

[ns311a] *Network simulator ns-3*, Online, http://www.nsnam.org, Dec 2011.

[ns311b] *ns-3.13 manual*, Online, http://www.nsnam.org/docs/release/3.13/manual/singlehtml/index.html, Dec 2011.

[PA00] V. Paxson and M. Allman, *Computing TCP's retransmission timer*, RFC 2988 (Proposed Standard), November 2000.

[PBRD03] C. Perkins, E. Belding-Royer, and S. Das, *Ad hoc On-Demand Distance Vector (AODV) Routing*, RFC 3561 (Proposed Standard), 2003.

[PH09] Guangyu Pei and Tom Henderson, *Validation of ns-3 802.11b phy model*, Online: http://www.nsnam.org/~pei/80211b.pdf, 2009.

[PH10] Guangyu Pei and Thomas Henderson, *Validation of OFDM error rate model in ns-3*, Online: http://www.nsnam.org/~pei/80211ofdm.pdf, 2010.

[PMSH10] Stylianos Papanastasiou, Jens Mittag, Erik G. Ström, and Hannes Hartenstein, *Bridging the gap between physical layer emulation and network simulation*, WCNC '10, 2010, pp. 1–6.

[PR10] David Peleg and Liam Roditty, *Localized spanner construction for ad hoc networks with variable transmission range*, ACM Trans. Sen. Netw. **7** (2010), no. 3, 25:1–25:14.

[PSB12] Bhawani Sankar Panda, D. Pushparaj Shetty, and Bijaya Kishor Bhatta, *Strong minimum energy minimum interference topology in wireless sensor networks*, ICDCIT, 2012, pp. 177–187.

[San05] Paolo Santi, *Topology control in wireless ad hoc and sensor networks*, Wiley, 2005.

[SKS06] Changsu Suh, Young-Bae Ko, and Dong-Min Son, *An energy efficient cross-layer MAC protocol for wireless sensor networks*, Proceedings of the 2006 international conference on Advanced Web and Network Technologies, and Applications (Berlin, Heidelberg), APWeb '06, Springer-Verlag, 2006, pp. 410–419.

[SWL04] Wen-Zhan Song, Yu Wang, and Xiang-Yang Li, *Localized algorithms for energy efficient topology in wireless ad hoc networks*, Proceedings of the 5nd ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc '04, ACM, 2004, pp. 98–108.

[vDL03] Tijs van Dam and Koen Langendoen, *An adaptive energy-efficient MAC protocol for wireless sensor networks*, Proceedings of the 1st international conference on Embedded networked sensor systems (New York, NY, USA), SenSys '03, ACM, 2003, pp. 171–180.

[VKW09] Markus Voelker, Bastian Katz, and Dorothea Wagner, *On the complexity of scheduling with power control in geometric SINR*, Technical Report 2009-15 (2009).

[vRWZ09] Pascal von Rickenbach, Roger Wattenhofer, and Aaron Zollinger, *Algorithmic models of interference in wireless ad hoc and sensor networks*, IEEE/ACM Trans. Netw. **17** (2009), no. 1, 172–185.

[WGG10] Klaus Wehrle, Mesut Günes, and James Gross (eds.), *Modeling and tools for network simulation*, Springer, 2010.

[WLBmW01] Roger Wattenhofer, Li Li, Paramvir Bahl, and Yi min Wang, *Distributed topology control for power efficient operation in multihop wireless ad hoc networks*, 2001, pp. 1388–1397.

[WW07] Dorothea Wagner and Roger Wattenhofer (eds.), *Algorithms for sensor and ad hoc networks*, Lecture Notes in Computer Science, vol. 4621, Springer, 2007.

[WZ03] Roger Wattenhofer and Aaron Zollinger, *XTC: A practical topology control algorithm for ad-hoc networks*, In 4th International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks, WMAN '03, 2003.

[Yao82] Andrew Chi-Chih Yao, *On constructing minimum spanning trees in k-dimensional spaces and related problems*, SIAM J. Comput. **11** (1982), no. 4, 721–736.

[YDE11] Onur Yilmaz, Orhan Dagdeviren, and Kayhan Erciyes, *Interference-aware dynamic algorithms for energy-efficient topology control in wireless ad hoc and sensor networks*, Comput. J. **54** (2011), no. 8, 1398–1411.

[YMG08] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal, *Wireless sensor network survey*, Computer Networks **52** (2008), no. 12, 2292–2330.