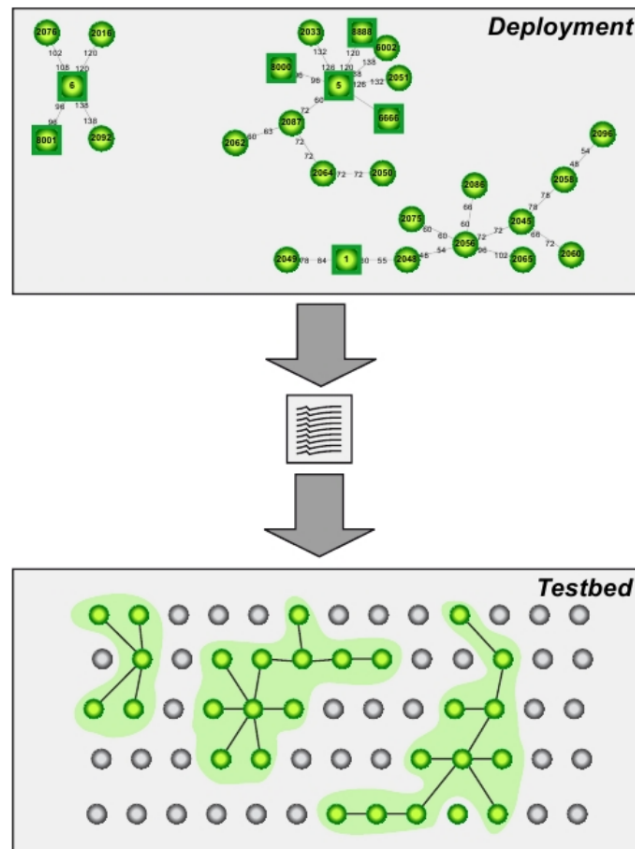


# Semester Thesis: Recipes for Wireless Sensor Networks

Andreas Marcaletti  
marandre@ee.ethz.ch

Supervisors: Roman Lim, Dr. Jan Beutel, Prof. Dr. Lothar Thiele



**Abstract**

This semester thesis deals with Wireless Sensor Networks (WSNs). The goal of the thesis is to develop a method for rebuilding deployed WSNs in a controlled testbed with high similarity. Such a method would be beneficial during planing and testing of new WSNs.

In order to achieve this we solved following problems. First we measured the testbed and created a dataset of links. Then we implemented an algorithm, that finds all the subnetworks available in testbed that have the same structure as the one we want to rebuild. Finally we applied a rating function to these subnetworks for finding the most similar one according to their link properties.

---

## Contents

1	Introduction . . . . .	4
1.1	Goal and Motivation . . . . .	4
1.2	Overview of the Thesis . . . . .	4
2	Related Work . . . . .	5
3	Measuring of the Testbed . . . . .	6
3.1	Values to Measure . . . . .	6
3.2	Testbed: FlockLab . . . . .	6
3.3	Hardware: Tmote Sky . . . . .	8
3.4	TinyOS Measurement Application . . . . .	9
	3.4.1 TinyOS . . . . .	9
	3.4.2 Increasing the Link Dataset . . . . .	9
	3.4.3 Description of the application . . . . .	9
	3.4.4 Management of the data . . . . .	10
4	Finding Similar Networks . . . . .	12
4.1	Ullmann Algorithm . . . . .	12
	4.1.1 Subgraph Isomorphisms . . . . .	12
	4.1.2 Description of the Algorithm . . . . .	13
4.2	Application of the Ullmann Algorithm . . . . .	14
	4.2.1 From the Link List to the Unweighted Directed Graph . . . . .	14
4.3	Rating . . . . .	14
5	Results . . . . .	17
5.1	Limits . . . . .	17
6	Conclusion . . . . .	19
6.1	Future Work . . . . .	19

## 1 Introduction

Wireless Sensor Networks (WSNs) consist small sensor nodes. These sensor nodes are capable of measuring the environment. For instance they can measure the temperature or the humidity. Besides that they have a radio interface which allows them to communicate with each other and form the wireless sensor network.

WSNs are used in numerous different scenarios. Some examples are

- For measuring the nature to predict earthquakes, tsunamis, avalanches, forest fires and so forth.
- For security systems in buildings.
- For traffic monitoring.

Benefits of WSNs are the flexibility, the low cost of deployment and maintenance. But WSNs are difficult, because they are distributed and the energy is constraint. Therefore, WSNs are a hot topic in research.

### 1.1 Goal and Motivation

The environment, where WSNs are deployed can be very harsh and not easily accessible, therefore it would be a benefit if one could once measure the potential WSN in the deployment and then rebuild in a controlled environment, a so called testbed. In a testbed often uses a backbone network to retrieve the data from the sensor nodes, therefore testing different protocols is easier in a testbed.

This semester thesis aims to develop a method, that allows to rebuild WSNs in a testbed. The possibility, that one can rebuild the network in every detail is very small, because WSNs are very complex. Therefore, this method should find the most similar network according to certain parameters.

For testing protocols this method could be useful as well. One could test protocols on numerous similar networks and investigate if the performance of these protocols is similar. Furthermore one could automatically generate certain network structures. If, for instance, the protocol is assumed to perform badly on a network with line topology, one could generate such networks to test the protocol on it.

### 1.2 Overview of the Thesis

We have split our problem into different subproblems. First we implemented a way to measure the testbed. This method should measure the link properties of the testbed to classify the quality of the links. The goal of that is to get a list of all the available links in the testbed. This method is described in section 3.

The second task is the actual matching. A description of the desired network has to be given so that we can look for the network that is the most similar to that one and available in the list links generated with the first method. This is described in section 4.

In section 5 we show some results of the implemented method and section 6 concludes the thesis and gives an outlook.

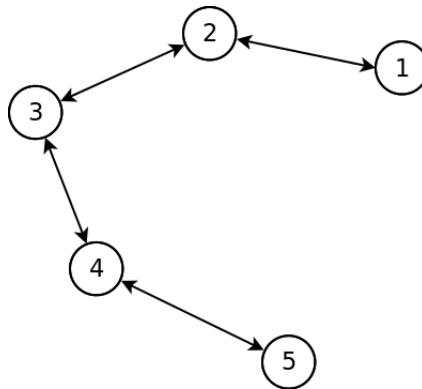


Fig. 1: Example of a WSN with line topology

## 2 Related Work

There are numerous different papers and thesis, that deal with measuring wireless sensor networks. In this section we want to highlight two of them.

In [1] they measured LQI and RSSI values of WSNs in different environments and compared these with each other. The used environments were a vineyard, a non operated tunnel and an operated tunnel. They found out that WSN can behave very different in these environments.

The goal of [2] was to build a testbed for wireless sensor networks. This testbed was especially designed for radio propagation measurements. Because they also measured the testbed in this thesis, it was a help for me to develop my measuring method. The idea of one node is broadcasting and the others are listening is from this thesis. The difference is, that in their thesis always the same node is broadcasting, while in our implementation we go through all nodes and let them broadcast one after another.

Finally, we have found in [3] an algorithm which we wanted to use to find similar networks. This algorithm is the Ullmann algorithm. It is further described in the subsection 4.1.

### 3 Measuring of the Testbed

In this section we describe the measuring of the testbed. We want to measure every available link in the testbed. With this information we are able to construct a network that resembles the desired network.

#### 3.1 Values to Measure

The values we want to measure are:

PRR: The Packet Reception Rate (PRR) is defined as follows:

$$PRR = \frac{\text{number of received packets}}{\text{number of transmitted packets}}.$$

It is a measure for the probability that a packet is successfully transmitted over this link.

SNR: Signal to Noise Ration specifies the difference in power between the power of the received packet and the background noise. It is defined as the ratio of the signal power and noise power. It is often specified in decibel.

RSSI: The Received Signal Strength Indicator (RSSI) is a measure for the power of the received signal. With this value and the power of the background noise one can calculate the Signal to Noise Ratio (SNR).

LQI: The Link Quality Indicator quantifies the link quality and is provided by IEEE 802.15.4 radios. We use the correlation value of the used Radio (see section3.3) as LQI which is in the range of 50 to 110. The higher the value, the better quality the link has.

#### 3.2 Testbed: FlockLab

In this section we describe the used testbed. We used a testbed developed and run by the Computer Engineering Group of the Computer Engineering and Network Laboratory at the Swiss Federal Institute of Technology Zurich called the FlockLab[4]. It is a WSN testbed that consists of 30 observer nodes. It has a mixed deployment with some indoor nodes and some outdoor ones. Every observer node consists of a embedded Linux Computer (Gumstix). They communicate over LAN or WLAN with each other. On every observer node one can mount up to 4 sensor nodes. Currently the supported sensor nodes are TinyNode, TmoteSky (TelosB), Opal nodes from CSIRO and Iris (MicaZ) nodes. The FlockLab is controlled by a central server, where the test data is collected. More about the architecture of the FlockLab can be found in [4].

The indoor nodes of FlockLab are located in an office building with interference sources like 802.11 WLANs and microwave ovens operating in the 2.4 GHz ISM Band, where the sensor nodes also operate.

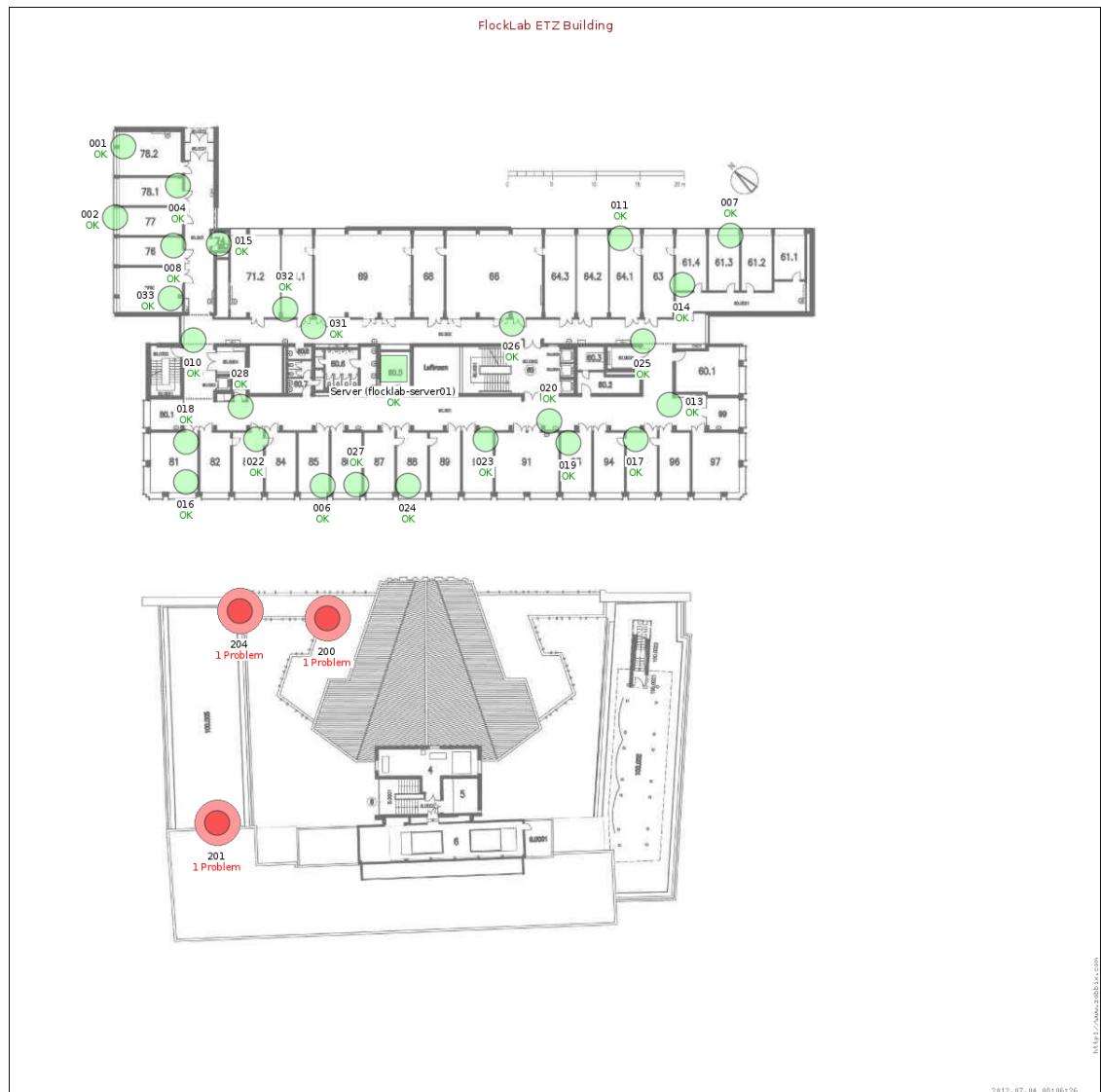


Fig. 2: Map of the FlockLab



Fig. 3: Picture of FlockLab observer node

### 3.3 Hardware: Tmote Sky

As mentioned one can mount up to four sensor nodes to a observer node in the FlockLab. For simplicity we decided to use only on sensor node platform, the Tmote Sky. Nevertheless the application can easily be ported to any other sensor node supported by FlockLab.

The Tmote Sky [5], also named TelosB, is a IEEE 802.15.4 compliant ultra low power sensor module. It is based on a TI MSP430 chip [7] and a Chipcon CC2420 radio [6]. It is developed by the company moteiv and its key features are:

- 250kbps 2.4 GHz IEEE 802.15.4 Chipcon Wireless Transceiver
- 8 MHz Texas Instruments MSP430 Micro controller
- 16 different channels in the 2.4 GHz ISM band
- Programmable transmit power in the range of 40 dBm
- On-board humidity, temperature and light sensors
- Ultra low current consumption
- Fast wakeup from sleep
- Programming and communication via USB
- Three programmable LEDs



## 3.4 TinyOS Measurement Application

### 3.4.1 TinyOS

The definition of TinyOS found on their website ([www.tinyos.net](http://www.tinyos.net)) is:

TinyOS is an open source, BSD-licensed operating system designed for low-power wireless devices, such as those used in sensor networks, ubiquitous computing, personal area networks, smart buildings, and smart meters.

It offers a well documented access to the all the functionality of the Tmote Sky. TinyOS applications are programmed in a dialect of C, called nesC. More about TinyOS and nesC can be found in [8].

### 3.4.2 Increasing the Link Dataset

We used three different transmit power levels and three different communication channels for measuring the testbed. The reason for using different power-levels and channels is, that with that approach we can measure more links and therefore we increase the dataset. The more links we have in the dataset, the higher is the possibility, that we are able to rebuild the same structure as the desired network.

We extended the used 23 nodes to 69 nodes thanks to the transmit power-levels and have three different datasets thanks to the communication channels.

To increase the link dataset even further, we measured the testbed at two different times of a day. One at midday, because then there is lot interference expected and one at midnight. Because at midnight no people are assumed to work in the building, where the testbed is mounted, therefore less interference is expected and the links have different characteristics.

### 3.4.3 Description of the application

The general idea is that one after the other every node transmits packets in a broadcast fashion and the rest of the nodes listen and collect the values mentioned in 3.1. As output data we generate messages in the sensor node and transmitted them via the serial reader to the observer nodes, then the data is recorded by FlockLab for every node. We make sure, that only one node at the



Fig. 4: Picture of Tmote Sky

time is transmitting in order to not introduce more interference by the sensor nodes itself.

The testbed gives the ability to trigger a GPIO signal of all sensor nodes simultaneously. We use this feature to establish a common point in time for the start of measuring.

The application is split in rounds. A round consists of the following:

1. All Nodes measure the background noise
2. Node 1 transmits its packets, while the rest of the nodes listens to the channel and captures the link properties.
3. Node 2 transmits its packets. And so on for the other nodes.
4. At the end once again every node measures the background noise.

In every round the communication takes place in a different communication channel. Now we describe the noise-measuring, the transmission of the packets and how the receiver reacts to received messages in more detail.

**Noise-Measuring:** The background noise is measured via the a provided TinyOS API. The measured value is between -5 and -55 dBm. We request the background noise level via this API 10'000 times and count how many times which noise level was measured. After these 10'000 scans we transmit for every noise level a serial message containing the number of times this scan level was measured.

**Transmitting:** As mentioned in 3.4.2, we used different transmit power levels. We decided to transmit 1000 packets per node and power level in every round. The power levels are interleaved as shown in figure 5: lowest power level, then middle, then highest and this is repeated 1000 times. The transmitted radio packet contains a counter, which is increment after every sent packet, the transmit power-level and some dummy payload, which is needed to make the measuring of the values more accurate.

**Receiving:** For every received packet a serial message is sent over the serial port to FlockLab. This message contains the RSSI value, the LQI value, the used communication channel and the transmit power-level. Because the serial port is slow compared to the radio interface we had to implement a buffer for the serial messages to make sure, that all message are sent to FlockLab.

#### 3.4.4 Management of the data

From FlockLab we receive a text file with all the recorded serial messages. We have to transform this into a list of links. This is done with a python script. This python script has the following functions:

- Reading all the serial messages
- Differencing between noise serial messages and value serial messages

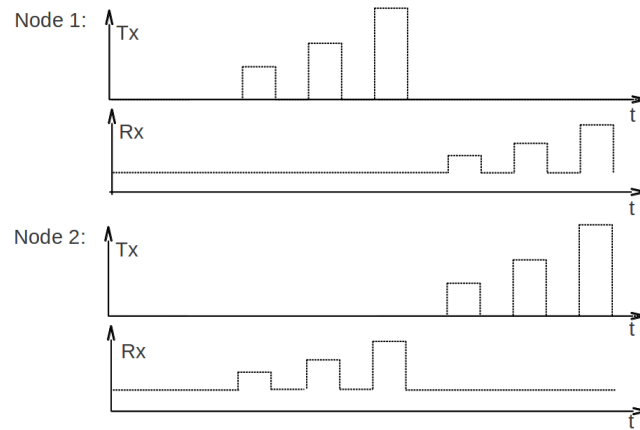


Fig. 5: Schematics of the measuring app

- Counting all the value serial message for and calculate the PRR for every link
- Storing a list of all the found links and their values
- Storing the noise data

As output we get two text files. One containing the link list and one containing the information about the noise.

## 4 Finding Similar Networks

In this section we describe our method that finds the most similar network. Similarity in the context of the WSNs is very vague. One could mean that the performance of different network protocols on the WSNs is similar. Can we predict such a similarity with using the captured link parameters described in section 3? We want to find a notion of similarity based on these values. Therefore we say the most similar network is the network with the same structure as the desired network and the smallest difference in the link properties.

Our approach to find this most similar network is split in two steps: First find all networks in your link dataset with the same structure and then rate all these networks to find the most similar one. In sections 4.1 and 4.2 we address the first problem with a graph matching algorithm and in 4.3 we describe the rating method for finding best matching.

Because we use a graph matching method we first have to transform the measured set of links into a graph. How we did this is described in section 4.2.1.

### 4.1 Ullmann Algorithm

The Ullmann Algorithm finds all subgraph isomorphisms of a graph  $G_\alpha$ , which represents the template network, in another graph  $G_\beta$ , which is a description of the link dataset. We use adjacency matrices  $A[a_{ij}]$  and  $B[b_{ij}]$  to describe the graphs. Both graphs are required to be unweighted graphs, therefore the matrices are defined as

$$a_{ij} = \begin{cases} 1, & \text{if there is a edge from node } i \text{ to node } j \text{ in the graph } G_\alpha \\ 0, & \text{else} \end{cases}$$

and

$$b_{ij} = \begin{cases} 1, & \text{if there is a edge from node } i \text{ to node } j \text{ in the graph } G_\beta \\ 0, & \text{else} \end{cases}.$$

#### 4.1.1 Subgraph Isomorphisms

If we now define a matrix  $M[m_{ij}]$  with the following properties:

- Same amount of rows as  $A$
- Same amount of columns as  $B$
- Every row contains exactly one 1, and the rest is zero
- Every column contains at most one 1

Now  $M$  represents a induced subgraph isomorphism if the following condition for the matrix  $C = M \cdot (M \cdot B)^T$  holds:

$$a_{ij} = c_{ij} \forall i, j. \quad (1)$$

If this conditions holds then all  $m_{ij} = 1$  mean that the  $i$ -th node in  $G_\alpha$  corresponds to the  $j$ -th node in  $G_\beta$ .

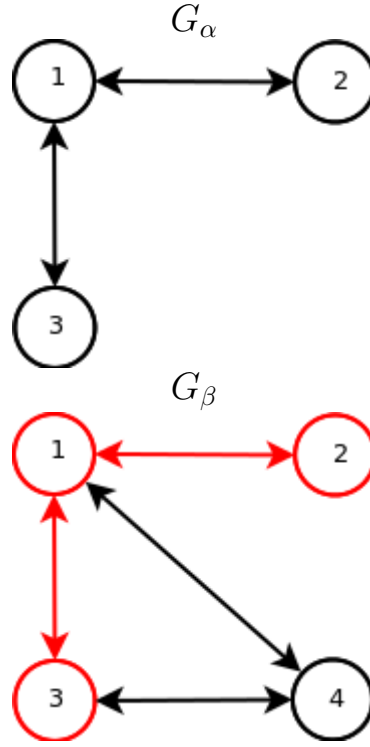


Fig. 6: Illustration of the isomorphism represented by  $M_1$  and  $M_2$

For example if we have  $A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$  and  $B = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$ , then  $M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$  and  $M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$  represent the subgraph isomorphism shown in figure 7.

#### 4.1.2 Description of the Algorithm

The Ullmann algorithm now tries find all the possible subgraph isomorphisms without generating all possible matrices  $M$ . It does that by choosing a clever start matrix. The start matrix  $M_0 [m_{ij}^{(0)}]$  is defined as follows

$$m_{ij}^{(0)} = \begin{cases} 0 & \text{if node } i \text{ in } G_\alpha \text{ has more edges, than node } j \text{ in } G_\beta \\ 1 & \text{else} \end{cases}$$

The algorithm cleverly shifts 1s in  $M_0$  to 0s in such a way that legitimate matrices  $M_k$  are created. These matrices  $M_k$  are now checked, with the condition (1), if they represent an isomorphism.

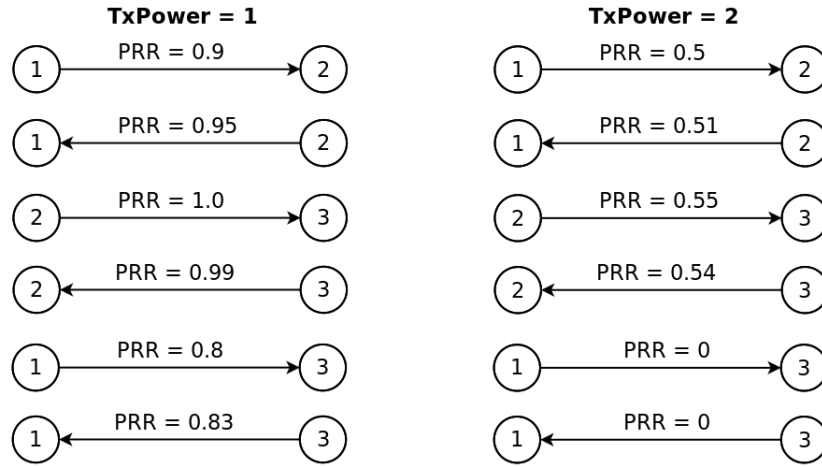


Fig. 7: Illustration of the link list

## 4.2 Application of the Ullmann Algorithm

We can now apply the Ullman algorithm if we represent the networks as unweighted graphs. Then  $G_\alpha$  represents the desired network and  $G_\beta$  the testbed.

### 4.2.1 From the Link List to the Unweighted Directed Graph

We have a link list as shown in figure 7. The algorithm needs an unweighted graph therefore we can use following condition for the edges.

There is an edge between node  $i$  and node  $j$  if  $PRR_{ij} > 0$

where  $PRR_{ij}$  stands for the PRR from node  $i$  to node  $j$ . We then have the graph shown in figure 9.

As mentioned in section 3 we captured links with different transmit power-levels. In a graph like in figure 9, where a node represents a sensor node with a certain transmit power-level we can add links from to nodes in other transmit levels because the transmit power does not affect the reception ability of a node. We then come to a graph as shown in figure 10. The blue edges represent the newly added ones.

In the graph shown in figure 10 every sensor node represented multiple times. We don't want to allow subgraph isomorphisms that include a node more than once with different transmit power-levels, because such nodes cannot exist at the same time and switching the power levels of a node would introduce a time dependency. To avoid this, we have to go through all the found subgraph isomorphisms and remove the solutions that violate this constraint.

## 4.3 Rating

If we use Ullmann's Algorithm for a graph as found in section 4.2.1 and a graph representing the desired network, we get numerous matrices  $M_k$  which define subgraph isomorphisms and therefore networks, that can be created in

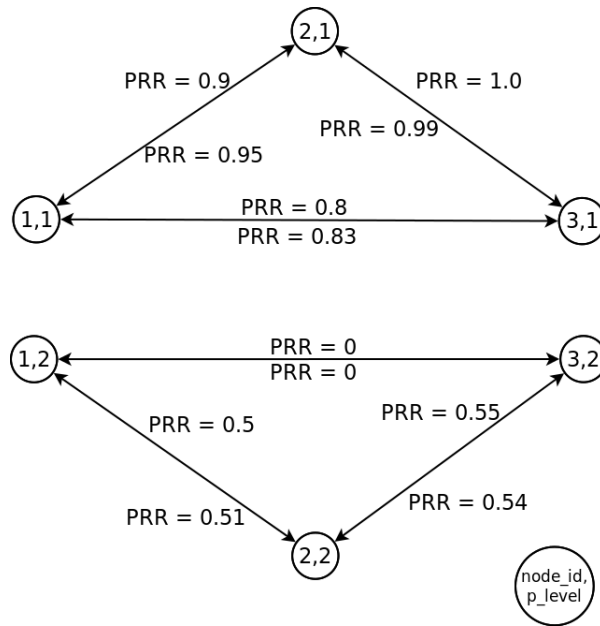


Fig. 8: Sensor network represented as a weighted graph. First index stands for the sensor node id and the second for the Tx power-level. Only PRR values are shown, SNR are omitted.

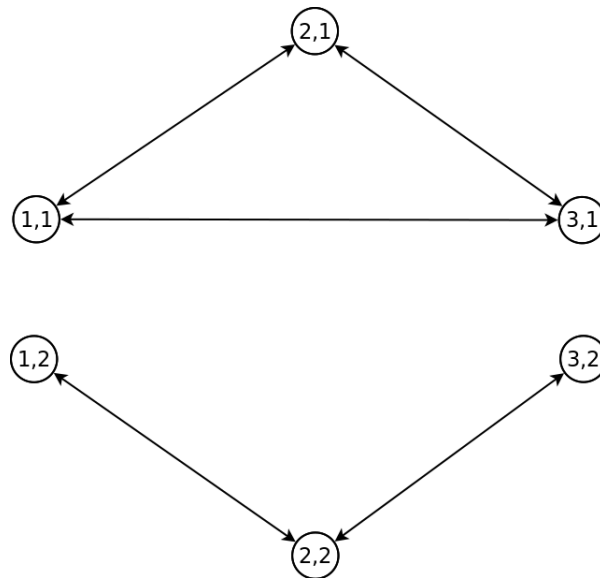


Fig. 9: Illustration of the unweighted graph. First index stands for the sensor node id and the second for the Tx power-level.

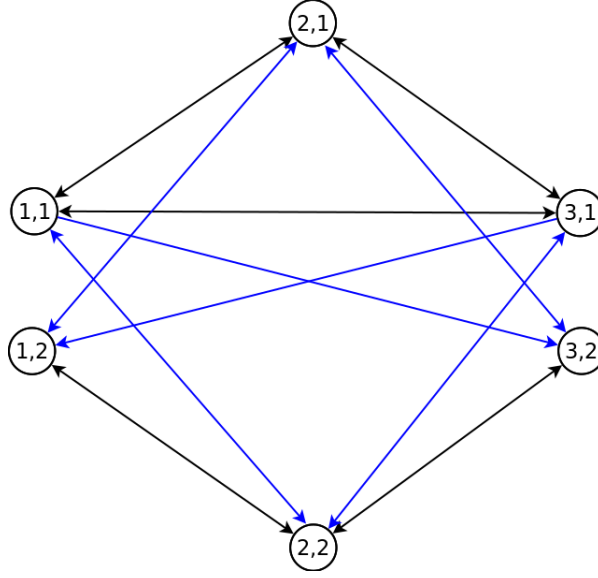


Fig. 10: Illustration of the final graph. First index stands for the sensor node id and the second for the Tx power-level. Blue edges are the ones added because of the Tx power-levels.

the testbed and have the same structure as the desired network. We now want to find the network that is the most similar of these. For that reason we compare the attributes of the links and calculate the following sum:

$$S = \sum_{l \in \text{All links in the networks}} s_l$$

with

$$s_l = \left| \frac{PRR_{l,f}}{PRR_{l,d}} - 1 \right| + \left| \frac{SNR_{l,f}}{SNR_{l,d}} - 1 \right|$$

Where the subscript  $l, d$  denotes the value of the link  $l$  of the desired network and  $l, f$  the value of the link  $l$  of the found network which we want to rate.  $S$  then represents the relative difference of the link values. Now the network with the smallest  $S$  is the best matching.

The rating function is a measuring of distance and can easily be changed. For instance one could also use the measured LQI values.



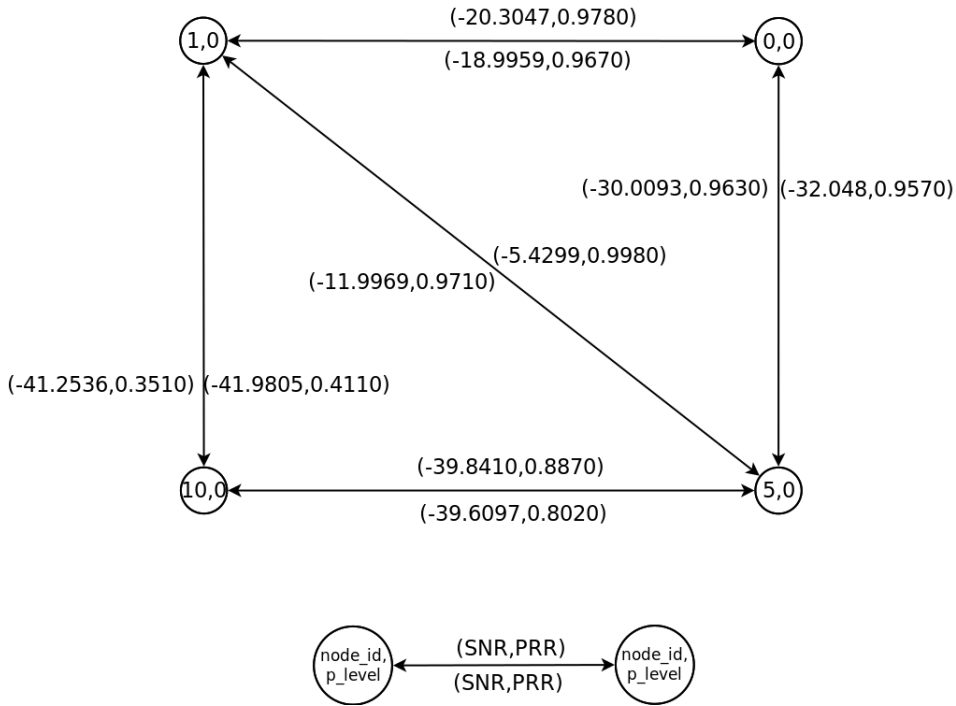


Fig. 11: Graph representing the given network. SNR is measured in dBm.

## 5 Results

In this section we want to present a result of the implemented method. We chose an existing subnetwork of the FlockLab and searched for the most similar network, which does not contain any of the node/ transmit power combination of the subnetwork. This network is illustrated in figure 11.

The best network is shown in figure 12 its rating result was  $S = 1.9730$ . The least similar with the same structure according to this method is shown in 13 and had a rating of  $S = 22.6237$

### 5.1 Limits

One limit of this method is the runtime. All test were performed on single Lenovo ThinkPad X201 with Intel i5 processor. The biggest impact on the runtime had the size of the network we want to find. While searching for 3 node network took several seconds, a 4 node network took several minutes and a 5 node network already several days. This is explainable by the fact that the number possible  $M_k$  increases exponentially by the number of nodes of the desired network.

Another limit is that this method can't be applied to every possible testbed. Networks build in the testbed should not vary to fast. If the variances of the attributes of the links in the testbed is to big we cannot rebuild the network at another time instance and the found network would only be valid at the exact time of the measuring.

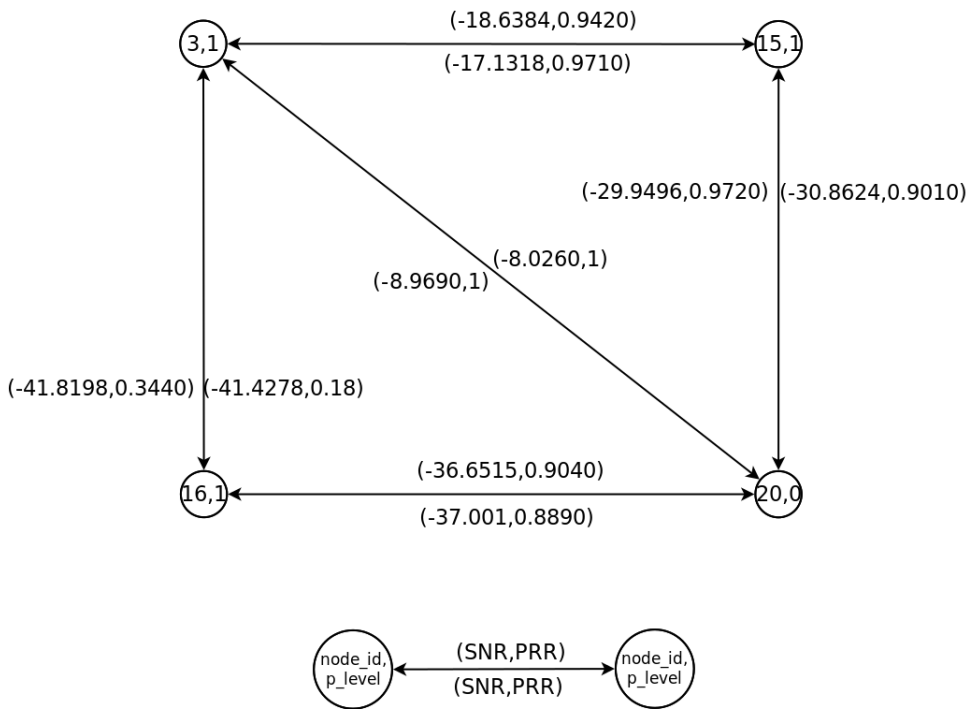


Fig. 12: Graph representing the most similar network. SNR is measured in dBm.

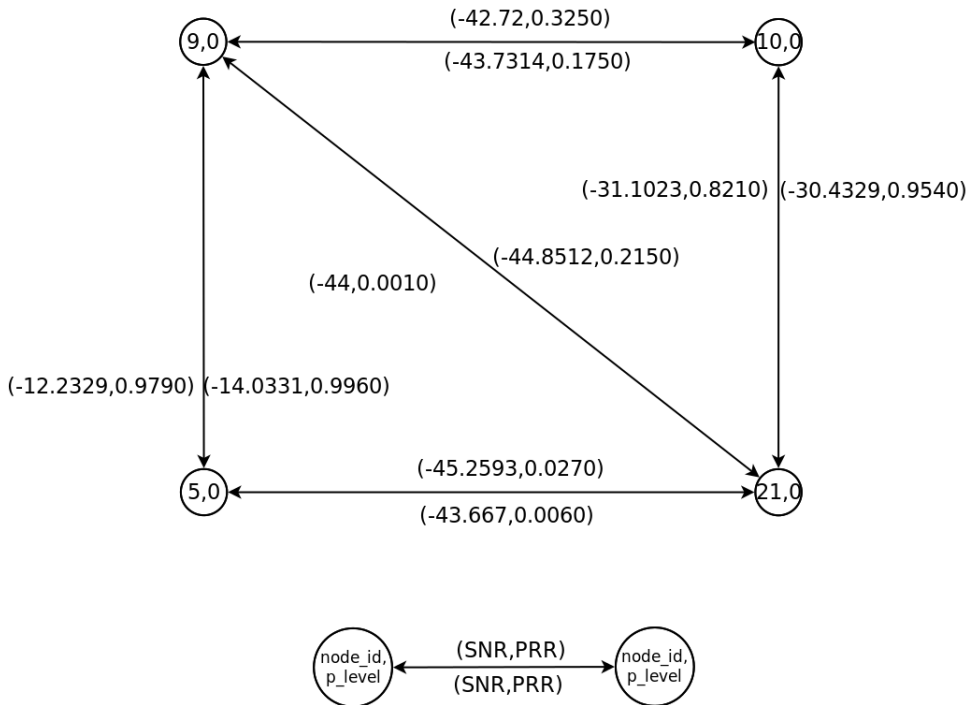


Fig. 13: Graph representing the least similar network with the same structure. SNR is measured in dBm.

## 6 Conclusion

In this thesis we managed to develop a method which allows building similar wireless sensor networks in a testbed. In order to achieve this we created a way to measure the testbed, adapted the Ullmann algorithm to our problem and developed a method for rating the found network, according to their link similarity.

### 6.1 Future Work

In this section we want to give an outlook on how this work can be used.

- The biggest drawback of this method is to run time. As mentioned in section 5.1 the application can run several days to find the best matching network. Therefore in future work one could speed up the Ullmann algorithm. One way to decrease the run time would be the following: One could modify the Ullmann Algorithm in way that it skips all matrices  $M_k$  that represent subnetworks which include the same sensor node twice with different transmit power levels. In our solution remove these networks after all are found.
- Develop new rating methods and compare them to each other. One could for instance compare the performance of standard protocols on the networks found with different rating methods. One could also include the dynamics of the links in the rating method by including the variance of the values measured.
- Use an inexact graph matching method. That means one would not look for the WSN with exactly same structure but rather look for a similar structure. After that one could compare performance of standard protocols on the networks found with exact matching and inexact matching.

## References

- [1] L. Mottola, G. P. Picco, M. Ceriotti, S. Guna, and A. L. Murphy, “Not all wireless sensor networks are created equal: A comparative study on tunnels,” *TOSN*, vol. 7, no. 2, 2010.
- [2] D. Burgener, “A Testbed for Radio Propagation Measurements in Wireless Sensor Networks”
- [3] J.R. Ullmann, “An Algorithm for Subgraph Isomorphisms”
- [4] J. Beutel, R. Lim, A. Meier, L. Thiele, C. Walser, M. Woehrle, and M. Yuecel, “The flocklab testbed architecture,” in *Proceedings of the 7th International Conference on Embedded Networked Sensor Systems, SenSys 2009, Berkeley, California, USA, November 4-6, 2009* (D. E. Culler, J. Liu, and M. Welsh, eds.), pp. 415–416, ACM, 2009.
- [5] Moteiv, “Data sheet: Ultra low power IEEE 802.15.4 compliant wireless sensor module”
- [6] Chipcon Products from Texas Instruments, “Data sheet: 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver”
- [7] Texas Instruments, “Data sheet: Mixed Signal Micro controller MSP430F2417”
- [8] Philip Levis, “TinyOS Programming”

## SEMESTERARBEIT

für  
Andreas MarcalettiBetreuer: Roman Lim  
Stellvertreter: Jan BeutelAusgabe: 7 Mai 2012  
Abgabe: 27 Juni 2012

---

**Recipes for Wireless Sensor Network Structures**

---

**Einleitung**

Ein drahtloses Sensornetzwerk besteht aus energielimitierten Knoten welche untereinander mit Funk kommunizieren. In den letzten Jahren wurde eine grosse Anzahl von Kommunikationsprotokollen entwickelt, meistens mit dem Ziel eine oder mehrere der folgenden Eigenschaften zu optimieren: Energieverbrauch, Durchsatz, Latenz und Datenverlust. Die Leistung der meisten Protokolle ist abhängig von der Topologie des Netzwerks.

Das Ziel dieser Semesterarbeit ist es eine Methode zu finden, mit der extrahierte Linklayer- und Topologie-Eigenschaften von existierenden drahtlosen Sensornetzwerken genutzt werden können um die gleiche Struktur in einem anderen Netzwerk nachzubilden, wie in Abbildung 1 dargestellt wird.

Dieser Ansatz ermöglicht es 1) Protokolle auf unterschiedlichen Testbeds mit ähnlichen Netzwerkstrukturen zu testen und so vergleichbare Ergebnisse zu erzielen und 2) kann diese Methode Vorteile bringen beim Planen von neuen Sensornetzwerken. In zweiten Fall wird im entsprechenden Gebiet ein Testnetzwerk installiert und Daten gesammelt. Anschliessend kann dann dieses Netzwerk auf einem Testbed nachgebildet werden und ermöglicht so eine sehr aussagekräftige Testumgebung.

**Aufgabenstellung**

1. Erstellen Sie einen Projektplan und legen Sie Meilensteine sowohl zeitlich wie auch thematisch fest. Insbesondere soll genügend Zeit für die Schlusspräsentation und den Bericht eingeplant werden.
2. Machen Sie sich mit den relevanten Arbeiten im Bereich Sensornetze vertraut. Führen Sie eine Literaturrecherche durch. Suchen Sie auch nach relevanten neueren Publikationen. Vergleichen Sie bestehende Konzepte anderer Universitäten. Prüfen Sie welche Ideen/Konzepte Sie aus diesen Lösungen verwenden können. Im Speziellen studieren sie [1], [2], [3] und [4].

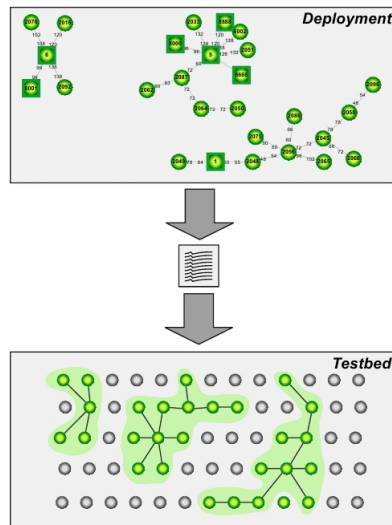


Abbildung 1: Die Struktur des Netzwerks wird extrahiert und auf die wesentlichen Merkmale reduziert. Anschliessend können diese Merkmale wie ein Rezept angewendet werden, um in einem Testbed eine ähnliche Struktur zu generieren.

3. Die Applikation soll auf dem Tmote Sky [5] entwickelt werden. Arbeiten Sie sich in die Softwareentwicklungsumgebung (TinyOS-2.x) der Knoten ein. Machen Sie sich mit den erforderlichen Tools vertraut und benutzen Sie die entsprechenden Hilfsmittel (Versionskontrolle, Bugtracker, online Dokumentation, Mailinglisten, Application Notes, Beispielapplikationen). Schauen Sie dazu insbesondere die TinyOS Webpage [6] an.
4. Als Testbed kommt FlockLab [7] zum Einsatz. Machen Sie sich vertraut mit dieser Testumgebung.
5. Erarbeiten Sie ein Konzept zur Sammlung der relevanten Netzwerkdaten (*PRR*, *RSSI*, *LQI*, *SNR*) und implementieren Sie eine Messanwendung auf dem Tmote Sky.
6. Werten Sie die gesammelten Daten aus und entwickeln Sie eine Methode mit welcher ähnliche Partitionen aus den in FlockLab vorhandenen Knoten erstellt werden können.
7. Evaluieren Sie die gefundenen Subnetze indem Sie die Charakteristik einer TinyOS-Standardanwendung (z.B. CTP [8] oder FTSP [9]) auf diesen Netzen vergleichen.
8. Dokumentieren Sie Ihre Arbeit sorgfältig mit einem Vortrag sowie mit einem Schlussbericht.

## Durchführung der Semesterarbeit

### Allgemeines

- Der Verlauf der Semesterarbeit soll laufend anhand des Projektplanes und der Meilensteine evaluiert werden. Unvorhergesehene Probleme beim eingeschlagenen Lösungsweg können Änderungen am Projektplan erforderlich machen. Diese sollen dokumentiert werden.
- Sie verfügen über einen PC mit Linux/Windows für Softwareentwicklung und Test. Für die Einhaltung der geltenden Sicherheitsrichtlinien der ETH Zürich sind Sie selbst verantwortlich. Falls damit Probleme auftauchen wenden Sie sich an Ihren Betreuer.
- Stellen Sie Ihr Projekt zu Beginn der Semesterarbeit in einem Kurzvortrag (maximal 5 Minuten) vor und präsentieren Sie die erarbeiteten Resultate am Schluss im Rahmen des Institutskolloquiums (maximal 15 Minuten).

- Besprechen Sie Ihr Vorgehen regelmässig mit Ihren Betreuern. Verfassen Sie dazu auch einen kurzen wöchentlichen Statusbericht (email).
- Sie führen ein Researchtagebuch in welchem Sie die Fortschritte täglich protokollieren.

## Abgabe

- Geben Sie zwei unterschriebene Exemplare des Berichtes, das Researchtagebuch sowie alle relevanten Source-, Object und Konfigurationsfiles bis spätestens am 27. Juni 2012 dem betreuenden Assistenten oder seinem Stellvertreter ab. Diese Aufgabenstellung soll im Bericht eingefügt werden, genauso wie das unterschriebene Unterschriftenblatt *Plagiat* des Rektorats. Die entsprechenden Richtlinien des Rektorats sind einzuhalten.
- Die Arbeit wird benotet anhand der Kriterien beschrieben in [10].
- Räumen Sie Ihre Rechnerkonten soweit auf, dass nur noch die relevanten Source- und Objectfiles, Konfigurationsfiles, benötigten Directorystrukturen usw. bestehen bleiben. Der Programmcode sowie die Filestruktur soll ausreichen dokumentiert sein. Eine spätere Anschlussarbeit soll auf dem hinterlassenen Stand aufbauen können.

## Literatur

- [1] L. Mottola, G. P. Picco, M. Ceriotti, S. Guna, and A. L. Murphy, "Not all wireless sensor networks are created equal: A comparative study on tunnels," *TOSN*, vol. 7, no. 2, 2010.
- [2] R. Maheshwari, S. Jain, and S. R. Das, "A measurement study of interference modeling and scheduling in low-power wireless networks," in *Proceedings of the 6th ACM conference on Embedded network sensor systems, SenSys '08*, (New York, NY, USA), pp. 141–154, ACM, 2008.
- [3] A. Marchiori, L. Guo, J. Thomas, and Q. Han, "Realistic performance analysis of wsn protocols through trace based simulation," in *Proceedings of the 7th ACM workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks, PE-WASUN '10*, (New York, NY, USA), pp. 87–94, ACM, 2010.
- [4] A. Meier, T. Rein, J. Beutel, and L. Thiele, "Coping with unreliable channels: Efficient link estimation for low-power wireless sensor networks," in *Proc. 5th Intl Conf. Networked Sensing Systems (INSS 2008)*, (Kanazawa, Japan), pp. 19–26, IEEE, June 2008.
- [5] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling ultra-low power wireless research," pp. 364–369, Apr. 2005.
- [6] D. Culler et al., "TinyOS: An operating system for Networked Sensors." <http://www.tinyos.net>.
- [7] J. Beutel, R. Lim, A. Meier, L. Thiele, C. Walser, M. Woehrle, and M. Yuecel, "The flocklab testbed architecture," in *Proceedings of the 7th International Conference on Embedded Networked Sensor Systems, SenSys 2009, Berkeley, California, USA, November 4-6, 2009* (D. E. Culler, J. Liu, and M. Welsh, eds.), pp. 415–416, ACM, 2009.
- [8] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proc. 7th ACM Conf. Embedded Networked Sensor Systems (SenSys 2009)*, pp. 1–14, 2009.
- [9] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi, "The flooding time synchronization protocol," in *Proceedings of the 2nd international conference on Embedded networked sensor systems, SenSys '04*, (New York, NY, USA), pp. 39–49, ACM, 2004.
- [10] TIK, "Notengebung bei Studien- und Diplomarbeiten." Computer Engineering and Networks Lab, ETH Zürich, Switzerland, May 1998.